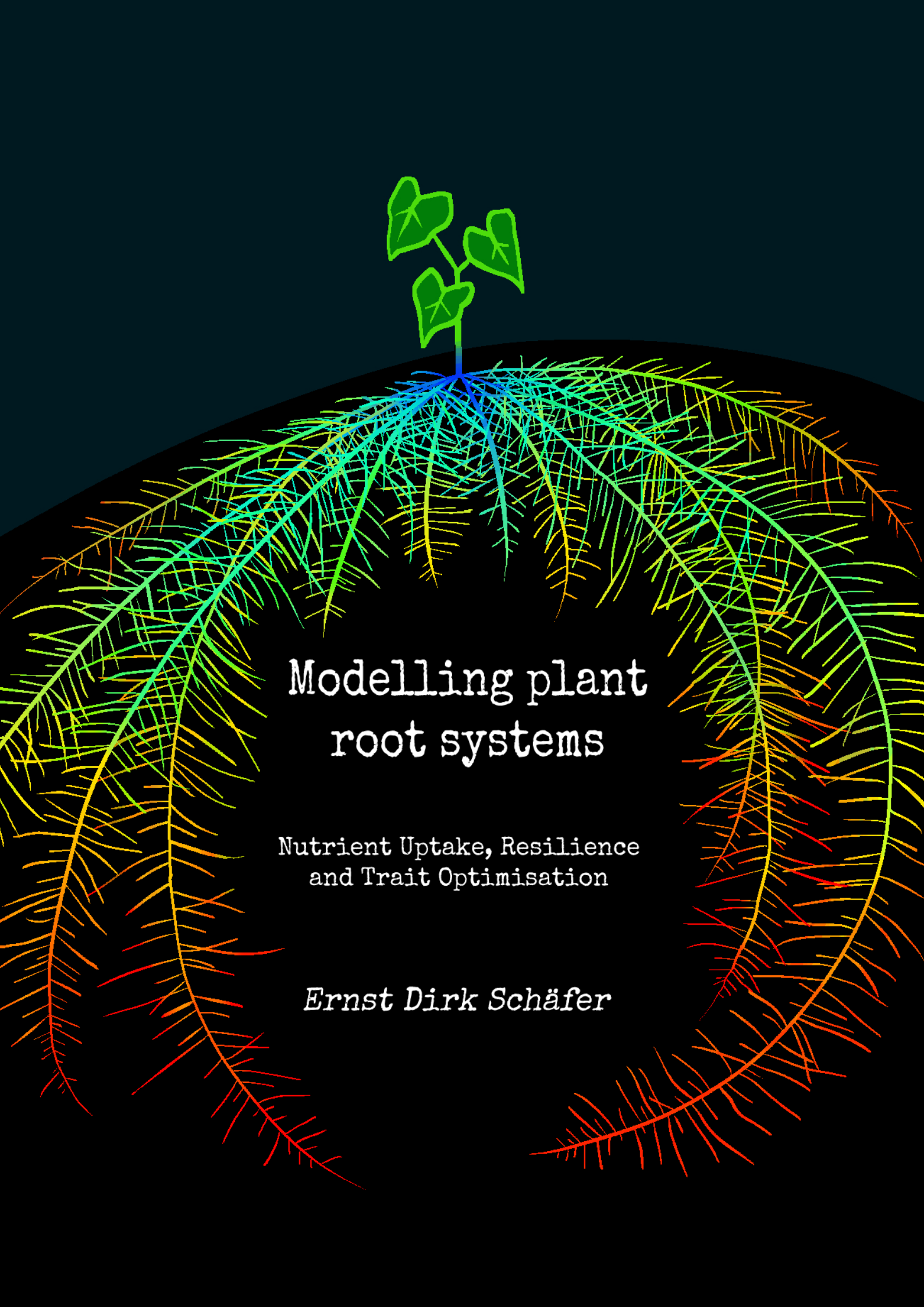


Modelling plant root systems

Nutrient Uptake, Resilience and Trait Optimisation

Ernst Dirk Schäfer

Modelling Plant Root Systems

Nutrient Uptake, Resilience and Trait Optimisation

Ernst Dirk Schäfer

supervised by:

Professor Markus R. Owen (Mathematics)

Professor Jonathan P. Lynch (Biosciences)

Professor Malcolm J. Bennett (Biosciences)

Doctor Leah R. Band (Biosciences)

Doctor Etienne Farcot (Mathematics)

Abstract

The green revolution led to a drastic increase in crop yields through chemical fertilisers, dwarf varieties and introduction of new methods of cultivation. We now face challenges such as climate change and soil degradation that require the development of crops resilient to adverse conditions in order to maintain an adequate food supply for the still growing world population. In order to maintain crop yields in soils with low nutrient availability or drought conditions we need to get a better understanding of the interactions between root system traits, soil environment and plant development. Compared to shoots, roots are hard to study because they are hidden from view by the soil. This makes mathematical models of roots a powerful tool to help us study root systems. We use OpenSimRoot, a functional-structural plant model to study the effects of various root system architectures on plant development in challenging environments, adding new functionality to expand the capabilities of OpenSimRoot. Our simulations showed that the effect of root loss on plant development depends on nutrient availability, plant species and root system phenotype, varying from very detrimental to slightly beneficial. Simulations of plants under drought implied that parsimonious and deeper rooting phenotypes perform better because of a large reduction in root carbon costs, increasing water uptake efficiency. We also show that machine learning techniques are a useful tool for root trait optimisation over a very large space of possible root system architectures. Our findings show that root system architecture has a large impact on plant development, especially in challenging environments and if we want to breed crops which are suited to deal with the challenges ahead of us we need to think about roots just as much as shoots. Our work also shows the benefits of interdisciplinary approaches by combining mathematical modelling with statistical machine learning in order to increase our understanding of biological systems. We hope our work will lead to increased collaborations across disciplines so that we may gain a better understanding of the hidden half of plants.

Acknowledgements

I am deeply grateful to Professor Markus Owen for his invaluable supervision, guidance in times of need and detailed feedback on all aspects my studies. I could not have wished for a better supervisor. I would also like to thank Doctor Leah Band for her digilent supervision and helping me work through difficult problems. I am grateful to Doctor Etienne Farcot for his patience in supervising me and for his invaluable advice. My gratitude extends to Professor Malcolm Bennett for supervising me and infecting me with his enthusiasm for science. I would like to express my sincere gratitude to Professor Jonathan Lynch for his guidance and supervision, for providing me with a new framework for thinking about biology and for pointing me towards interesting questions. I am also grateful to Doctor Nathan Mellor for helping me get started with OpenSimRoot. I would like to thank Doctor Johannes Postma, Christian Kuppe and Doctor Chris Black for helping to refine my understanding of OpenSimRoot. I would like to thank Doctor Ishan Ajmera for working with me on implementing and testing drought related models. I am also grateful to Doctor Ian Vernon for sharing his statistical expertise and guiding me through complicated mathematics. I would like to thank my friends and colleagues from the Modelling and Analytics for a Sustainable Society programme for the good times we shared both in and outside the office. My appreciation goes out to my family, and especially my parents for their support and encouragement during all of my studies. Special thanks go out to Lae Schäfer, who created the beautiful cover art. I am grateful for access to the University of Nottingham High Performance Computing Facility. Finally, I would like to thank the Leverhulme Trust for funding my studies.

Declaration

I declare that all the work in this thesis is mine and not submitted before for a degree. Whenever contributions of others are involved, every effort is made to indicate this clearly, with due reference to the literature. I collaborated with Doctor Ishan Ajmera on Chapter 4, we decided which models to use together, he provided model parameters and we discussed and tested the models together. Doctor Ian Vernon provided expert advice and guidance for Chapter 5, making sure every step of the process was statistically sound and carefully considered.

Some of the research described in this thesis is being prepared for publication. A chapter on OpenSimRoot was written for a book on methods in biology (currently being finalised). Chapter 3 is a draft paper which is under revision and will be resubmitted. A publication about the research described in Chapter 4 is in preparation and work has started on several joint papers in collaboration with various researchers at Penn State University. Some more simulations and analysis are planned on the work described in Chapter 5, after which this will be written up for publication.

The work described in this thesis was done under the guidance of Professor Markus Owen at the School of Mathematical Sciences, University of Nottingham.

Contents

Chapter 1

Introduction

Plants form the basis of the terrestrial ecosystem. Energy, in the form of sunlight, enters the ecosystem through the process of photosynthesis, which fixes carbon into energy carrier molecules. Through herbivory, these molecules enter the animal kingdom and allow the existence of other forms of life. Humanity realised the importance of plants for nutrition thousands of years ago and when some hunter gatherers transitioned to a sedentary agricultural lifestyle our way of life was about to fundamentally change. While the genetic mechanisms would no be uncovered for thousands of years, early farmers understood how to increase yields through selective breeding and the resulting food surpluses meant that it was no longer a necessity for everyone to spend their time producing food. This lead to the emergence of new occupations and technologies and set the stage for our modern civilisation. Through the ages agricultural yields have increased because of improvements in irrigation, soil management practices, selective breeding and better tools. The most recent agricultural revolution happened between the 1940s and the 1960s, the so-called Green Revolution which led to massive increases in crop productivity and preceded a population boom. The Green Revolution drastically changed agriculture around the world, making extensive fertilizer and pesticide use a common practice in most parts of the world.

Because of unsustainable farming practices, soil quality is degrading in many parts of the world. Fertiliser and irrigation use can partially address this, but these are expensive, energy intensive, pollute the surrounding environment and are not feasible for many farmers, especially in Africa and South-East Asia. Climate change is expected to exacerbate the deterioration of soil quality as well

as lead to more floods and droughts. This threatens food security, especially because the human population is predicted to increase to 9.6 billion by 2050. Thus the challenge we face is to ensure food security despite the deterioration of soil and the shift in environmental conditions that climate change is expected to bring. Because the environmental changes will strongly impact nutrient and water availability, it seems clear that we should increase our understanding of how plants acquire water and nutrients in a variety of environments. Evidently the place to start our investigation is with the organs responsible for the uptake of water and nutrients, the roots.

Since roots are (mostly) underground, they are more difficult to study than other parts of the plant. Until recently, one had to dig up the roots to study them, irrevocably altering their environment and often destroying the more delicate parts of the root system in the process. With the advent of computers and modern imaging techniques, new avenues to study roots have been opened up. In this thesis, we will use a computer simulation model, OpenSimRoot [156], to study the relation between root traits, water and nutrient uptake and crop productivity. We do this with the aim of increasing crop productivity in environments with low nutrient and water availability.

1.1 Thesis overview

Chapter 1 is this introduction. Chapter 2 provides an overview of relevant plant science research and concepts. It describes soils, nutrient and water uptake, photosynthesis, functions of roots, root system architecture and methods of studying roots. Finally, it provides a general overview of root system architecture modelling and a brief description of OpenSimRoot and some important models in OpenSimRoot. Chapter 3 is a draft paper describing an OpenSimRoot study on the effect of root loss on barley, bean and maize in a number of different environments. A root loss module was developed for OpenSimRoot for this purpose. Chapter 4 describes a number of models related to photosynthesis and plant water status that we added to OpenSimRoot. These models enable modelling of C3 and C4 photosynthesis and stomatal responses to drought in OpenSimRoot, which was not possible before. The results of an OpenSimRoot study with these models comparing two different phenotypes under well-watered and drought conditions are laid out. Chapter 5 lays out an emulation-based approach that was used to determine the root architectural parameters associated with the highest maize shoot dry weight for plants grown in a soil low in nitrogen, phosphorus and potassium. It highlights how statistical machine learning techniques can be used to get a grip on high-dimensional parameter spaces for models with long running times such as OpenSimRoot. Appendix A contains supplementary material relating to Chapter 3. Appendix B contains tables of symbols, constants and variables related to the drought model of Chapter 4 and various tables related to Chapter 5. Appendix C contains a descriptions of a useful numerical optimisation used in Chapter 5. Appendix D contains a comprehensive guide to getting started with running OpenSimRoot and developing new functionality.

Chapter 2

Literature Review

Plants are vital for life on earth. Plants take up carbon from the air in the form of CO_2 and use photosynthesis to produce the carbohydrates that are important energy carrier molecules in most biological organisms. These carbohydrates are used for growth, respiration and reproduction or stored for later use. All animals depend, directly or indirectly, on these carbohydrates to survive and reproduce. But it is not just because of these carbohydrates that plants are essential for terrestrial life on earth. Plants also contain the mineral nutrients that organisms need to function. Most plants acquire these mineral nutrients, along with water needed for photosynthesis and transpiration, from the soil through their root system. Apart from this essential task, roots fulfill other tasks, such as providing anchorage for the plant or storing water and/or nutrients. In light of this, it is not surprising that most plants invest considerable resources in their root systems. This is evidenced by the fact that the dry weight (the weight after removing all water by drying) of the roots can be larger than the dry weight of the shoot for a range of species, including trees, which have a considerable aboveground presence [32, 64, 105]. It is not just the relative size of the root system that highlights its importance, considerable resources are invested in maintaing roots as well. More than a fifth of the carbon plants produce by photosynthesis can be spent on root respiration [8, 150]. Plants subjected to nutrient stress often shift their biomass allocation in favour of roots and under these conditions they can spend as much as 50% of their daily assimilated carbon on respiration [138, 101]. Plants can also spend a significant fraction (8-17%) of the carbon they produce [81] on root secretions. These secretions, called root exudates, serve a multitude of purposes, such as providing a lubricant that helps the roots penetrate the soil

[127], enhancing the acquisition of immobile nutrients such as phosphate [124, 143], helping the plant defend against pathogens [10, 206] and altering the pH value [15] and microbiome of the rhizosphere [11].

Roots are also an integral component of the soil carbon cycle. Plants take up carbon in the form of CO_2 from the air and use this to produce carbohydrates (sugars). This is then used for respiration, growth, reproduction, or stored for later use. Plants, as well as animals, return the carbon to the atmosphere when they combine the carbohydrates with oxygen to release the chemical energy contained in it. There are of course many more components in the carbon cycle, as it involves all life on land, in the seas and in the soil. The soil is estimated to contain three times more organic carbon than the vegetation that grows on it, making it the largest reservoir of organic carbon in terrestrial ecosystems [153]. Many organisms in the soil rely on this supply of organic carbon. This is especially true for the organisms in the rhizosphere, which is the region of soil directly influenced by root secretions. Plant detritus is another major source of soil carbon, with roots contributing more than shoots, especially in deeper soil layers [65, 165]. Cereals, for example, are estimated to sequester about 1500 kg C ha^{-1} yr^{-1} into the soil [99]. It is important to note that changes in agricultural practices and rooting patterns could potentially increase the amount of carbon sequestered in the soil by about 300 to 800 kg C ha^{-1} yr^{-1} increasing the total amount of C stored permanently in the soil by 50 tonnes C ha^{-1} [94]. This could form a significant component of the efforts to mitigate the severity of climate change.

2.1 Photosynthesis

Photosynthesis is the process by which plants fix energy from sunlight into carbohydrates. In the process, carbon dioxide (CO_2) is fixed from the atmosphere and oxygen (O_2) is often (but not always) released as a waste product. For a more comprehensive overview, see [14, 141]. We will give an overview of the processes most relevant to this thesis.

Photosynthesis in plants requires light, carbon dioxide and water, as well as the

proteins involved in the process, most notably chlorophyll, and nutrients required to replenish those. Light is provided by the sun and absorbed by chlorophyll pigments in the leaves. Since chlorophyll does not absorb green light, leaves appear green to us. Absorbed light excites electrons which starts a flow of electrons down an electron transport chain, a series of reactions that transfers electrons from donor to acceptor molecules. In these light-dependent reactions, ATP and NADPH, energy carrying molecules, are produced some of which are later used in the light-independent reactions. Relevant for our purposes is the fact that light is needed at two steps in the electron transport chain, and is absorbed by protein complexes called photosystem 1 and 2 (PSI and PSII).

Carbon dioxide enters leaves through stomata, pores in the epidermis (the outer layer of tissue) of leaves through which gas exchange happens. A pair of specialised cells called guard cells are situated at each stomata, which allows the plant to control the rate of gas exchange. Through the stomata, carbon dioxide enters substomatal chambers from which it can diffuse into what are called mesophyll cells. In C3 plants, which fix carbon through the Calvin cycle, this is where carbon is fixed by the enzyme RuBisCO (Ribulose-1,5-bisphosphate carboxylase-oxygenase). This enzyme catalyses the carboxylation (a reaction which produces a carboxylic acid from a substrate and carbon dioxide) or RuBP (ribulose-1,5-bisphosphate). RuBisCO has the unfortunate property that it also catalyses the oxygenation of RuBP, which is called photorespiration. Photorespiration is a wasteful process that creates toxic waste products that need to be detoxified, which costs valuable energy. Because of this, photosynthesis is most efficient when there is a high CO_2 concentration at the site of carboxylation. In C3 plants this requires a high stomatal aperture which allows CO_2 to enter the mesophyll cells easily. This comes at a cost, since it also means that water diffuses out of the stomata easily. While some water is needed in the photosynthesis reaction, this is negligible in comparison to the amount of water lost to transpiration. The water loss due to transpiration creates a low hydraulic potential which allows the plant to transport water and nutrients from the roots up to the shoot. Transpiration also lowers leaf temperature, this can help prevent heat stress from damaging leaf tissues. If water is in short supply, plants reduce stomatal aperture to limit water loss, but this also reduces photosynthesis rates because mesophyll CO_2

concentrations will decrease.

While the majority of plant species on earth use C3 photosynthesis, there are other pathways used by a small number of species. One of these is the C4 pathway, used by, for example, maize, sugarcane and millet. The C4 pathway is an example of convergent evolution, having evolved up to 61 times in different plant families. Plants that use this pathway temporarily fix carbon in mesophyll cells to PEP (phosphoenolpyruvate), which is then transported to what are called bundle sheath cells. Here carbon is fixed into sugars by RuBisCO. This allows C4 plants to fix more CO_2 with the same amount of transpiration, their water use efficiency is higher. At high temperatures, the oxygen affinity for RuBisCo increases, which increases the advantages of C4 over C3 photosynthesis as temperatures increase. However, because of the extra steps involved, C4 plants are less efficient when water is plentiful since they need to spend energy carboxylating and regenerating PEP. C4 plants are also more nitrogen efficient because PEP carboxylase requires less nitrogen than RuBisCO. C4 plants have an advantage when water and nitrogen are limiting but light is plentiful and temperatures are high, while C3 plants have an advantage when light is limiting, water and nitrogen are readily available and temperatures are low.

2.2 Soils

We provide a brief overview of relevant soil science here, for a more comprehensive introduction, see [79]. Soil is generally divided into three layers, the topsoil, subsoil and parent material. The parent material generally is bedrock that the other soil layers rest on. Plants grow in the topsoil and subsoil layers. Soils consist of mineral particles, organic matter and contains gaps (pore spaces) that are filled with water and air. The relative amounts of these components determine the physical properties of the soil, as well as its suitability for plant growth. The mineral component of soil consists of rock particles of varying sizes, called sand, silt and clay. These terms refer to particles of different size and the relative amounts of these determine the texture of the soil. Sand particles are the largest, ranging from 0.05 to 2.0 mm. Silt particles range from 0.002 to 0.05 mm while particles smaller than 0.002 mm are the smallest and make up the clay particles.

Clay particles consist of different minerals than sand and silt and has a platelike structure, as opposed to the more spherical shapes of sand and silt particles. The relative proportions of these three types of particles determine the soil type, as shown in Figure 2.1.

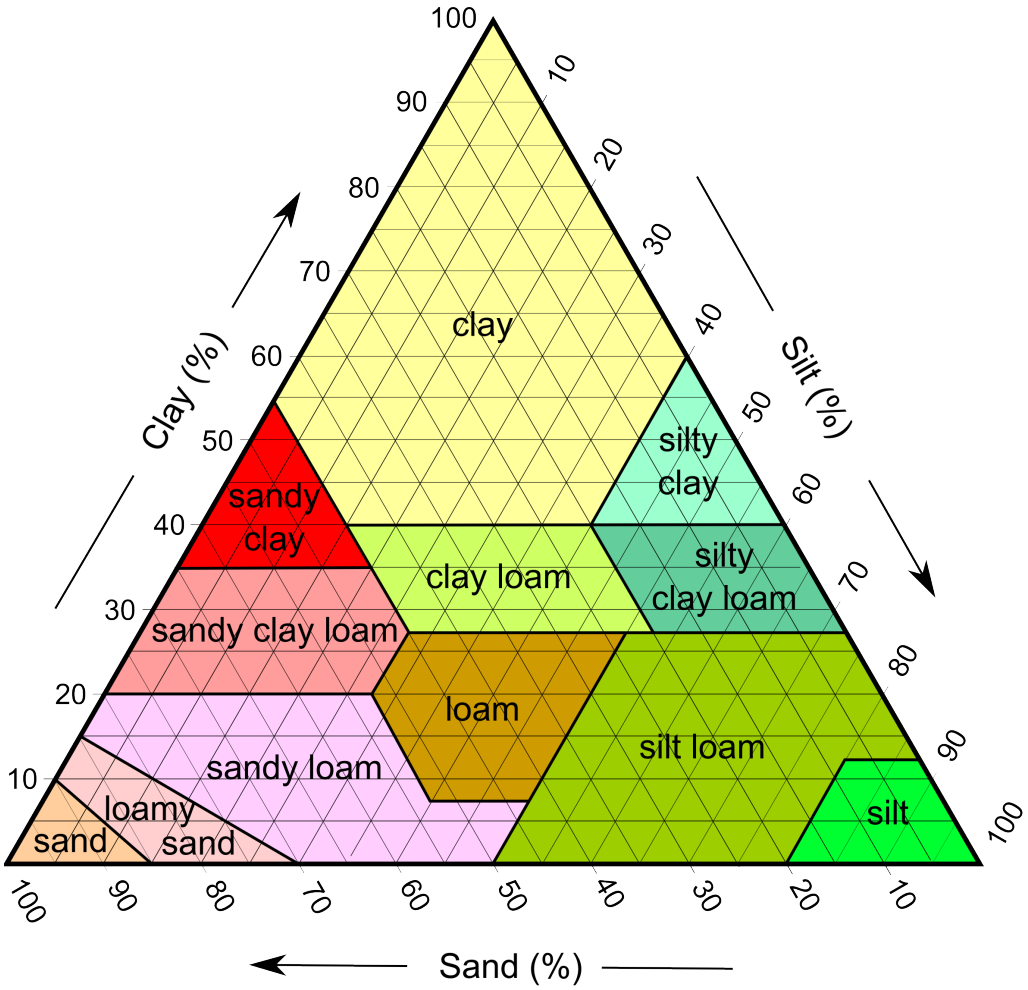


Figure 2.1: A soil texture diagram of soil types according to their clay, silt and sand fractions [168].

As a soil particle decreases in size, the ratio between surface area and volume increases. This means that soils consisting of smaller particles have a larger specific surface area (surface area per gram). Soils with a large specific surface area can hold more water and nutrients and often contain higher amounts of organic matter. Sandy soils are unable to retain water or nutrients for long and are hence difficult to cultivate. Clay soils present other difficulties in that they are very hard when dry and very sticky when wet. Soils with high specific surface area have a high buffer power for certain nutrients, which prevents nutrient concentrations

from declining rapidly.

The pore structure of a soil also affects the ability to retain water. Pores are grouped into three categories, macro (large), meso (medium) and micro (small) pores. Macropores drain rapidly because the adhesion and cohesion forces are not sufficient to overcome the gravitational pull exerted on the water. Soil compaction decreases the amount of pores and the pore sizes and detrimentally affect a soil's capacity to retain water. It also prevents water from being absorbed by the soil, leading to drainage problems. Apart from soil compaction, there are other processes that threaten soil quality and fertility. Topsoil erosion occurs naturally and is usually a slow process but certain land management practices can increase the rate of erosion. Rain can cause soil erosion, especially when reduced filtration leads to surface water runoff. Wind erodes soils by blowing soil particles away. Tillage leads to erosion, by exposing soil to water and wind erosion and causing soil particles to move downslope. Worldwide, many topsoils are subject to erosion, decreasing fertility, as can be seen in Figure 2.2. Vegetation can protect soils from erosion, by shielding the soil from the direct impact of rain, providing cover from wind and increasing the capacity of soils to take up water. Cover crops are often used to shield soils outside the growing season.

Most soils contain organic matter, which is composed of deceased plants and animals. It contains many different soil organisms such as bacteria, fungi and insects, which consume the organic matter and/or plant roots for sustenance. These organisms break down the organic matter and release CO_2, which increases the acidity of the soil. The CO_2 reacts with soil minerals and releases nutrients in forms that can be taken up by plants. The amount of organic matter in a soil depends on the amount of precipitation, the temperature and the drainage rate of the soil, amongst other factors. Soils that drain quickly, have a large pore fraction or are tilled frequently tend to contain more air which increases the organic matter decomposition rate. Soils with poor drainage have a low oxygen content which limits the decomposition of organic materials.

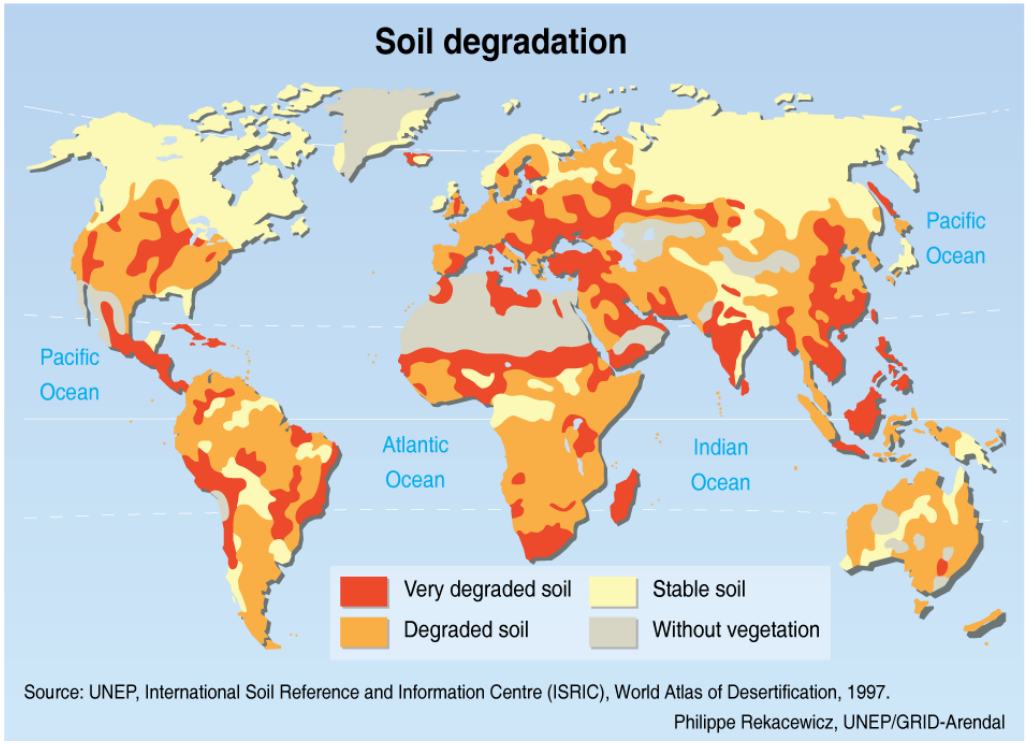


Figure 2.2: Soil degradation around the world [166].

As described above, the pores in a soil are important in determining the water retention capacity of a soil. There are typically three forces acting on water molecules in the soil: gravity, cohesion forces and adhesion forces. Gravity pulls water downwards through the soil and causes soils to drain over time. Cohesion forces between water molecules, are stronger than the cohesion forces between most other substances because water molecules form hydrogen bonds. The strong cohesion of water molecules creates a surface tension. Adhesion is the attraction between water molecules and solid surfaces, such as soil particles. Water also exhibits capillary action, which allows it to move up narrow tubes. Capillary action happens when the adhesion forces of a fluid to a solid tube exceeds the cohesion forces within the fluid. A concave meniscus forms and the surface tension pulls the fluid upwards. Capillary action allows the soil to retain water and move it upwards or horizontally. Finer textured soils have smaller pore sizes and thus have stronger capillary action.

2.3 Nutrients

We summarise the basics of plant nutrition here, for a more comprehensive overview see [12]. There are three criteria that determine if a nutrient is essential:

- The plant is unable to complete a regular life cycle in the absence of this element.
- No other element is able to completely substitute for the element.
- The element is part of an essential plant constituent or metabolite or the element is required for the activity of an essential enxyme.

Following these criteria, there are 18 elements that are essential for plants. They are generally divided into macronutrients and micronutrients. Macronutrients are present in large quantities in plants and make up the bulk of the dry weight of plants while micronutrients make up less than 0.02% of plant dry weight and the presence of micronutrients in plant tissues is typically measured in parts per million. Macronutrients are further subdivided into structural, primary and secondary nutrients. The structural macronutrients are carbon, hydrogen and oxygen. The primary macronutrients are nitrogen, phosphorus and potassium. The secondary macronutrients are calcium, sulfur and magnesium. The micronutrients are iron, boron, chlorine, manganese, zinc, copper, molybdenum, nickel and cobalt. Hydrogen, oxygen and carbon are obtained from air and water while the other macro and micronutrients are acquired from the soil by most plant species. The Sprengel-Liebig law of the minimum states that crop growth is limited by the scarcest resource [148]. Because of this, maximum crop yield will only be attained if all essential elements are sufficiently available, in the right proportions.

Of the macronutrients acquired from the soil, nitrogen is needed in the largest quantities and often limits crop productivity in the absence of fertilizers. While dinitrogen (N_2) forms about 78% of the earth's atmosphere, only a few species have evolved the ability to fix nitrogen from the air. Nitrogen exists in many different forms but most plants can only take up inorganic nitrogen, ammonium (NH_4^+) and/or nitrate (NO_3^-), directly. This makes understanding the nitrogen

cycle an important component of understanding plant functioning in different environments. The most important transformations of nitrogen are fixation, mineralisation, immobilisation, nitrification, denitrification, volatilisation and leaching. We will discuss each of them briefly.

- Fixation is the process that converts atmospheric nitrogen (N_2) to ammonium (NH_4^+). Some bacteria can convert atmospheric nitrogen to ammonium and several plant species, such as legumes and rice, are able to form a symbiotic relationship with nitrogen-fixing organisms. The Haber-Bosch technique is a chemical process that fixes nitrogen from the atmosphere, however, this requires large amounts of fossil fuels. Nitrogen also enters the soil through abiotic processes such as acid rain and lightning (lightning breaks the strong molecular bonds between nitrogen atoms in N_2 after which these nitrogen atoms react with oxygen and water, forming NO_x, HNO_3 and NO_3^-).

- Mineralisation is the conversion of organic forms of nitrogen into ammonium by microbes. The rate of conversion depends on the amount of organic matter in the soil, the temperature, the oxygen content, soil moisture content and the amount of carbon in the soil.

- Immobilisation is the conversion of inorganic nitrogen, ammonium (NH_4^+) and/or nitrate (NO_3^-), into organic nitrogen, the reverse of mineralisation. This often happens if soil organic matter is high in carbon but low in nitrogen.

- Nitrification is the conversion of ammonium (NH_4^+) to nitrate (NO_3^-) by soil organisms. Ammonium is first converted to nitrite (NO_2^-), which is poisonous to plants, and then to nitrate.

- Denitrification is the conversion of soil nitrate (NO_3^-) to atmospheric dinitrogen (N_2) by soil organisms. This only occurs when carbon is present and oxygen is absent.

- Volatilisation is the loss of ammonium (NH_4^+) through conversion to ammonia gas (NH_3) which is released to the atmosphere. Volatisation can lead to considerable nitrogen losses when soil pH rises above 7.5 and temperatures are high enough.

- Leaching is the process by which substances drain out of soil with the water. While ammonium is retained well by the soil and thus mostly immobile, nitrate is highly mobile and can leach out of the soil easily. Apart from drastically lowering the nitrate content of the soil, this can cause damage to the environment.

The second important primary nutrient is phosphorus. It is needed in smaller quantities than nitrate but often a limiting resource, especially in the tropics. Phosphorus is usually taken up as $H_2PO_4^-$ or HPO_4^{2-} but some organic forms of phosphorus can also be taken up by plants. Like the nitrate cycle, the phosphorus cycle is complex and consists of many different processes. Unlike many other cycles, the atmosphere is not a significant component of the phosphorus cycle. Phosphorus tends to bind tightly to soil particles, through a process called sorption. This means that in many soils, a significant fraction of the phosphorus is not found in the soil solution but in unavailable, bound forms. Because of this, phosphorus is called "buffered". The processes of precipitation, when phosphorus forms a solid mineral in reaction with other substances, and dissolution, when phosphate minerals dissolve and release phosphorus, are both important for the phosporus cycle, though they are both very slow processes. Like nitrate, phosphorus is subject to mineralisation and immobilisation. Because phosphorus tends to bind tightly to soil particles, it is less mobile and less likely to leach out of the soil. This also means it is often concentrated in the topsoil. Most plants, including maize, barley, wheat and rice, form symbiotic relationships with mycorrhizal fungi that form long and thin hyphae and enhance phosphorus uptake.

2.4 Root functioning

See [132] for a more comprehensive overview of the mechamisms behind plant uptake of water and nutrients. Water is vital for plants, just as it is vital for any living organism. Plants take up water by generating a water potential (which is the potential energy of water at a reference pressure, temperature and elevation) in their roots that is lower than the water potential in the surrounding soil. They have to overcome the capillary and adhesive forces that bind water to the soil. The primary process by which plants do this is by evaporating water in the shoot. The shoot is hydraulically connected to the roots through the xylem vessels that

run through the plant tissue and this creates a low water pressure potential in the roots. Furthermore the osmotic potential of root cells is generally lower than the osmotic potential of the soil solution and this difference in osmotic potential causes water to move into the roots. There are also active processes that help absorb water from the soil. Once water has entered the root it is transported inwards to the xylem vessels which carry it upwards to the shoot. The most widely accepted explanation of how plants are able to transport water to canopies that can be up to 100 metres above the soil surface is the cohesion-tension mechanism: menisci at the air-water interface in leaves are exposed to evaporation; the surface tension pulls water molecules into the locations occupied by molecules that have evaporated; because of the strong cohesion in the xylem sap, this pull is transmitted along the continuous water column all the way down to the roots.

Plants have developed a wide variety of strategies that increase their ability to take up nutrients from the soil. As mentioned before, nutrients are often found in the form of charged particles and are often electrostatically bound to the soil. To displace them from the soil, many plants exude protons (H^+). This is especially important for immobile nutrients such as phosphorus, most of which is often bound to the soil. Roots also contain transport proteins that aid in nutrient uptake. Passive transporters act as channels that allow substances that typically cannot pass through the cell wall and membrane to enter the roots. Active transporters are able to move solutes against concentration gradients, but consume energy in the process.

Plants have developed several strategies that increase the surface area of their root-soil interface. Root hairs are long and thin outgrowths formed by cells in the root epidermis above the elongation zone. Root hairs have a high surface to volume ratio because of their long and thin structure and they allow plants to explore a much larger fraction of the soil, this makes them very useful for the uptake of immobile nutrients such as phosphorus. Another strategy that increases the fraction of soil that a plant can explore is the formation of a symbiotic relationship with mycorrhizal fungi. Over 80% of all plant families form mycorrhizae and some plants depend on them for development [23]. The plant supplies the fungus with carbohydrates and in return the fungus provides the plant with water

and nutrients. The fungus does this by forming hyphae, long root-like filaments, that are able to explore large parts of the soil that the plant would not reach by itself. Some fungi also have the ability to mobilize mineral nutrients from the soil through releasing chemicals.

A different form of symbiosis allows plants to fix nitrogen from the air. Only a limited number of species, including legumes, can form this symbiotic relationship with nitrogen-fixing bacteria, called rhizobia. The symbiosis is initiated when the plant releases a chemical, which signals that it is looking for rhizobia [206]. Rhizobia respond by releasing nodulation factors, which prompts the plant to form a specialised type of root hairs that allow the rhizobia to enter the root tissue. Cells start dividing to form a root nodule containing the rhizobia, that fix atmospheric nitrogen for the plant, in exchange for carbohydrates. Plants that can form this symbiotic relationship can be used to increase the soil nitrogen levels.

We mentioned root exudation, the secretion of various molecules by roots, before. Root exudates do not only serve to make nutrients better available for uptake, they fullfill a variety of other functions. The region of soil that is directly influenced by root exudates is called the rhizosphere. In the rhizosphere root exudates change soil pH to facilitate nutrient uptake, neutralise toxic compounds, regulate soil microbiota, encourage beneficial symbioses, inhibit growth of competing plant species and release toxins against pathogens [198, 206]. There is still much we do not understand about the interactions between root exudates and soil micro-organisms. A recent study found that gases emitted by roots can influence soil microbia several centimetres away from the roots [151]. For the purposes of this thesis, we are mostly interested in exudates as a process with a carbon cost.

Waterlogged plants commonly develop (root cortical) aerenchyma (RCA), enlarged gas spaces in root tissue [57, 213]. They allow for the transport of oxygen to the roots which would otherwise suffer from anoxic conditions and transport gases from the soil to the atmosphere. Carbon dioxide produced by roots and soil organisms is one of the gases transported up by aerenchyma, as is methane, an important greenhouse gas. It is estimated that more than 80% of methane emitted

from waterlogged soils reaches the atmosphere via aerenchyma [28]. Aerenchyma can also form in dryland species such as barley, maize and wheat as a result of normal development or stress. Aerenchyma can reduce radial nutrient transport into roots because nutrients can not travel across the air-filled cavities [84] but also decreases root respiration carbon requirements, providing an advantage in certain environments [36, 159].

2.5 Root System Architecture and its Importance

Architecture plays an important part in root functioning [111, 113]. Root architecture, while mostly determined by genotype, is known to change in response to environmental factors [108, 154]. This indicates that different architectural adaptations are needed for differing environments. One major environmental constraint that plants have to adapt to is spatial and temporal inhomogeneity that soils often exhibit. This includes variation in soil type and structure as well as nutrient and water content [90]. This inhomogeneity is due to the natural processes of soil morphogenesis, natural processes like wetting, drying and temperature changes, the activity of organisms like earthworms and the roots themselves. A consequence of this inhomogeneity is that a plant can potentially save a lot of resources by growing roots where they will be most efficient and avoiding nutrient-poor or dry patches of soil.

Indeed, plants have developed adaptations to heterogeneous conditions. An often observed response to nutrient-rich patches is a local increase in root proliferation [170, 171]. Another example is the observation that low phosphorous availability leads common bean plants to grow a more shallow root system, while a deeper root system is grown in response to water stress [80]. As the previous example illustrates, different environmental stresses can lead to opposite responses, indicating that plants have to find a balance between extremes. A better understanding of the stresses and external conditions a plant experiences and the costs and benefits of architectural adaptations to these will allow us to find yield-improving phenotypes adapted to different environments more efficiently.
Having considered the importance of roots and root architecture we shall now

provide some key definitions:

- **Morphology:** The shape or surface features of roots as organs. This includes features like root hairs, the root diameter, the root length, the distribution of lateral roots amongst others. The morphology can be determined by inspection of the root in question.

- **Topology:** The way in which roots are connected to each other, ignoring the precise spatial configuration (modulo deformations and rotations). This definition coincides with the mathematical one and as such only tells us which roots subtend from each other. The topology can be determined through any method of measuring which keeps roots intact.

- **Distribution:** The average spatial distribution of roots in the soil. Knowledge of the distribution provides you with the expected values for root length density or root biomass per volume. The distribution can be measured by taking soil samples and measuring the amount of root length or root biomass.

- **Architecture:** The precise spatial configuration of the root system; how roots are connected and their locations in the soil. Usually, root hairs and other fine details are not taken into account when studying root architecture. Measuring architecture requires careful measurement the position of each root, as well as its connections to other roots, either through nondestructive imaging or careful excavation of the root system.

It is clear that knowledge of root system architecture implies knowledge of the topology and distribution. Topology and distribution are generally good proxies for architecture but can not fully capture it [111]. The following terms are often used to characterise root systems [9]:

- **Embryonic root:** Roots derived from the embryo (seed). Primary and seminal roots (see below) make up the embryonic roots. Roots which form after germination are referred to as postembryonic roots.

- **Primary root:** The first root to appear from the seed as it germinates is called the primary root or tap root (see below). The primary root is also referred to as radicle.

- **Tap root:** A large primary root from which all the other roots emerge laterally is called a tap root. Tap roots can develop into storage organs such as is the case for carrots.
- **Seminal root:** The embryonic roots, excluding the primary root, are called seminal roots. Sometimes the primary root is also counted among the seminal roots.
- **Adventitious root:** Any postembryonic root which does not develop from root tissue is called an adventitious root.
- **Hypocotyl:** The stem of a germinating plant. It connects the roots to the rest of the plant.
- **Nodal root:** The shoot consists of nodes, which hold leaves and buds that can grow into branches, and internodes, that separate the nodes. Roots that emerge on shoot nodes are called nodal roots.
- **Crown root:** Nodal roots that emerge on belowground shoot nodes are called crown roots.
- **Brace root:** Nodal roots that emerge on aboveground shoot nodes are called brace roots. They are also called stilt or prop roots.
- **Basal root:** Roots that emerge along the base of the hypocotyl are called basal roots.
- **Order:** The order of a root is equal to the order of the root it branched from plus one. Roots emerging directly from the seed or from non-root tissue have order one.
- **Lateral root:** This can refer to any root that branches off another root. They can also be called branch root, secondary root, tertiary root, depending on the order.
- **Axial root:** We use axial root to refer to any non-lateral root. This includes primary, tap, seminal, adventitious, nodal, crown and basal roots.
- **Root hair:** A tubular outgrowth of an epidermis cell, only found near the tips of roots. Root hairs take up water and nutrients from the soil.
- **Terminal root:** A root from which no other roots have branched.

It is important to note that there are no universally accepted naming conventions and that conventions tend to vary depending on the plant under consideration. We have chosen nomenclatures that are often used when discussing barley, bean and maize, the species relevant to this thesis.

In this thesis we will refer to the totality of a plant's observable characteristics as the phenotype of that plant [218, 217]. Every phenotype is composed of phenes and the value a phene has in a given phenotype is the phene state. So a phene is to a phenotype, what a gene is to a genotype, that is, it is a distinct element of a phenotype. Different genotypes may result in the same phenotype. Examples of phenes are the number of nodal roots, the angle at which certain roots emerge and the lateral branching density. Phenes can be synergistic or not and many architectural phenes will contribute to the overall phenotype of a root system.

A clear example of the importance of root system architecture is provided by the need for a plant to obtain immobile resources that are concentrated in the top soil (phosphorus, potassium) while also obtaining mobile resources that tend to leach down into the soil (water, nitrate). The optimal phenotypes for addressing these two challenges are very different: Shallow and dense root systems are better at collecting immobile resources from the top soil, while deep and sparse root systems are better suited to collecting mobile resources or those in deep soil. Thus a plant is faced with a trade-off and has to find a balance between these extremes that allows it to perform both functions in heterogeneous and often uncertain environments. For example, the optimal lateral branching density depends on nitrogen and phosphorus concentrations, with low branching densities being better when nitrogen is limiting and high branching densities being better when phosphorus is limiting [154].

Competition between plants also has implications for the optimal root system architecture. If a nutrient is scarce, the plant which can take it up the fastest will outperform others and by doing this can even impede the development of neighboring plants. It was shown that shallow-rooted bean root systems offered a competitive advantage if phosphorus was concentrated in the topsoil [172]. In a sense, this is analogous to how competition for sunlight results in plants growing

taller, all in an effort to escape the shadow of their competitors. Because of the dynamics between environment and the competition with neighbouring plants, the optimal strategy can be difficult to determine.

Many plant root systems show plasticity in reaction to environmental cues. The most obvious plastic response observed in plants is the variability of the shoot/-root ratio. It has been observed that nutrient and water stress tends to decrease the shoot/root ratio, while soils with ample supply of both tend to increase the shoot/root ratio [5, 18, 13, 212]. This is a form of negative feedback that allows a plant to adapt its strategy to increase the production or uptake of the most limiting resource.

Other forms of plasticity follow the same pattern; a plant tries to adapt such that limiting resources are acquired with as little investment of carbon and nutrients as possible. Localized root proliferation in response to nutrient-rich patches is an example of a mechanism that allows plants to efficiently allocate resources [170, 171]. Plants also exhibit hydrotropism, which means they respond to water gradients, allowing their roots to grow towards water [52]. These adaptive growth patterns allow plants to function in a wider range of environments and their existence proves that they are at least somewhat beneficial.

2.6 Root Loss

Plants are constantly under threat of herbivory, disease and nutrient deficiency. Their root systems are not exempt from these threats and root loss is prevalent in many species. Total root production in sugar beet, winter barley and winter wheat was observed to exceed standing root system size by a factor two to four, which means that 50 to 75 percent of produced root length was lost [184, 188, 185].

Root loss research has mostly been focused on trees or was done in the context of ecology and there has been relatively little research into the effects of root loss on plant fitness and crop productivity. Studying roots is more challenging than studying shoots, by virtue of roots being underground. The effect of root loss on plants is difficult to discern from field experiments because it can not be

controlled directly. The effects of environmental factors on root loss has been studied and the most prominent factors that affect root loss are:

- **Herbivores and pathogens.** There are many organisms in the soil, some of which feed on roots. Examples include root-knot nematodes, moulds and root mealybugs [93, 142, 209]. Applying pesticides to roots increases their lifespan considerably [208]. This seems to indicate that while root longevity is, to some extent, under control of the plant, herbivores or pathogens are often a direct cause for the loss of roots.

- **Soil moisture.** There is limited information on the effect of drought on root loss [16]. Tomato root length density in a saline soil was up to 40% lower [180] and the effect of drought on the mortality of fine roots and roots of species without an endodermis is substantial [75]. On the other hand, citrus roots restrict the allocation of carbon to the root and slow root respiration under drought. This slows the metabolism of the cells in the roots, leading to severely restricted nutrient uptake, because the active transporters involved in uptake require energy to function [24]. Once the drought is over, the roots essentially recover completely in a short amount of time [54].

- **Soil temperature.** Temperature affects root production, but its effects are not always well characterised. King et al. found that lowering the soil temperature has no clear effect on the root longevity of trembling aspen [96]. On the other hand, increasing soil temperature has been found to increase root mortality in grasses [63], white clover [207] and sugar maple [77]. The mean annual temperature was the most important variable explaining fine-root turnover in the global data set on root turnover of Gill and Jackson [71], and it suggests a mean annual temperature increase of 10°C leads to a 40-90% decrease in root lifespan. There are of course a lot of variables that vary with temperature and this makes it difficult to ascertain if the effect on root longevity is a direct consequence of the temperature difference or indirectly through pathogen activity or soil quality. It should also be noted that some studies found that increasing the temperature had no clear effect on root mortality [87, 183].

- **Soil nutrients.** The availability of soil nutrients seems to influence root

lifespan but there are conflicting results: some studies found that high nutrient availability coincides with low root longevity [123, 161, 162, 184], while others have found the opposite [1, 26, 160].

- **Mycorrhizal fungi.** Mycorrhizal associations seem to protect roots from a variety of factors, enhancing their longevity [56, 67, 137].

The uptake rates, of both water and nutrients, of a root generally decline over time. This is because

1. As a root takes up nutrients, the surrounding soil is depleted of water and nutrients.

2. In many species, root maturation coincides with a decrease in radial hydraulic conductivity, lowering the uptake capacity.

This reduction in uptake rates then lowers the amount of water and nutrients gained per amount of energy (in the form of carbohydrates) invested in root maintenance. Plants that spend the carbon (energy) they produce more effectively, will of course be more successful so it is clear that plants will try to maximise the efficiency of their root system. The carbon cost of the respiration necessary to maintain a root can outgrow the carbon cost of growing the root in as little as 20 days [55]. It follows from the above facts, that there are situations where maintaining the existing roots is less effective than investing in new roots. Some plants have indeed adopted a strategy of shedding fine roots under drought and regrowing them when rains arrive, thus avoiding the costs of maintaining a root system when it is not needed [140].

However, respiration rates generally decline with increasing root age as well (after secondary growth has subsided) because cells stop growing and dividing, the cells are generally bigger in a mature root (so there are less cells per root volume) and aerenchyma formation and root cortical senescence can lower the number of cells. This means that even when uptake rates decline, roots might become more efficient over time. One expects yield to be optimised by plants that maximise the cumulative uptake efficiency of their roots. Under the condition that the root respiration rates decline at a sufficiently low rate and assuming the soil is depleted of nutrients, it can be shown there is some finite time at which the cumulative

uptake efficiency of the roots is maximized [21, 214]. Based on this, one would expect programmed root senescence to be common, but evidence for this is lacking [61]. Root length does generally decrease during and after flowering, but this can simply be because plants stop investing in new roots during this stage of their development [55].

2.7 Experimental approaches to studying plant roots and root loss

Studying roots is much more difficult than studying shoots or leaves, because they are obscured by the soil. We will discuss a number of different techniques for studying roots which can be categorised into: excavation methods, in-situ monitoring methods and labeling methods. Excavation methods, like field coring and shovelomics, involve the removal of all, or parts of, the root system from the soil. This means only one measurement can be taken, making these methods suboptimal to study root loss. Non destructive monitoring techniques leave the root system intact and try to minimise disruption, though, as we will see, some deviation from normal growth conditions is inevitable. Labelling methods are based on the placement of certain isotopes (for example ^{15}N) of nutrients in specific locations in the soil. Information about the root system can then be deduced from the concentrations of these isotopes that are detected in the shoot. A good overview of most of the methods discussed here can be found in [17] and [149].

- **Excavation methods.**
  - **Root system excavation.** A trench is dug at an appropriate distance from the root system. Then, layer by layer, the soil is removed carefully so as not to damage any roots [17]. During the excavation, the position and size of roots can be documented using cameras. A lot of different parameters can be determined using this method though biomass is often underestimated because of the sometimes considerable root losses during excavation.
  - **Shovelomics.** Probably the most straightforward way to obtain information about root systems is by doing shovelomics, which works as

follows: Using a shovel, a certain soil volume around the base of the shoot is dug up. Then the soil is washed off the root system in soapy water after which analysis on the roots is possible [193]. While several traits such as the number and angle of crown and brace roots can be measured, the method is only applicable to small root systems. Both very fine roots and the relative positions of roots can be lost in the process.

  - **Soil coring.** Soil coring is a useful method to determine the root length density in a given soil volume. A metal pipe lined with a plastic tube is pounded into the soil. It is then taken out and the plastic tube is removed. The cylinder of soil in this plastic tube is taken out and the root segments in this cylinder are separated from the soil by dissolving it in soapy water and using a sieve to gather the root segments [199]. This method is useful to compare measures like the root length density, the biomass density or the root tip density between plants. Care must be taken to choose representative samples at consistent locations relative to the plants sampled.

- **Non destructive monitoring methods.**

  - **Root window or rhizotron.** A glass or plexiglass window is pressed against the soil profile, this allows one to study the roots in their 'natural' environment [53]. Many root system traits can be studied using this technique, subject to some caveats. The window should be installed in a representative place, so some prior knowledge of the root distribution is necessary. Obviously, each window only allows one to study one single slice of the soil. The window will of course impede root growth and might alter the growth pattern.
  - **Minirhizotron.** This technique is useful to assess the production and turnover of roots. A transparent tube is inserted into a hole drilled in the soil. A camera or mirror is lowered into the tube and images spanning the tube are recorded. Repeating this will give insight into the growth and turnover rates of the root system [35]. See [61] for results obtained with a minirhizotron and [122] for a discussion of the technique and a comparison with the soil coring and ingrowth core technique.

– **Non destructive imaging.** Using MRI or CT scanning equipment, one can get three-dimensional images of root systems grown in columns, see Figure 2.3. The only constraints here are the size of the columns, which necessarily are too small to fully contain most root systems, and the resolution of the scanner. Depending on this, fine laterals will or will not be visible. Significant progress has been made since 2011, when it was demonstrated this technique could be of use [135]. Image analysis software is required to distinguish between soil and root, which, considering the host of different soil particles that can usually be found in the soil, can be quite a challenge [121]. While this method can not always be applied to plants grown in natural conditions, it has the potential to give the most extensive and accurate information.

It should be clear that each of these methods has severe limitations and either requires extensive preparations, a lot of effort or expensive, specialised equipment. This highlights the importance of modelling as an important tool to study roots, both as an explanatory tool as well as a provider of interesting research directions for experiments.

2.8 Root System Architecture Models

Mathematical models are simplified representations of reality that aim to accurately reproduce processes or dynamics. They can be used to predict the effect of changing parameters or environmental variables, provide insight into the mechanisms that lead to observed behaviour and point the way towards new questions. In an agronomical context, mathematical models are an important scientific tool. Molecular and cellular models are used to gain insight into the (regulation of) mechanisms driving important processes such as photosynthesis, germination, root branching and responding to environmental changes. Climate models are used to predict the effects of possible climate change scenarios on local weather conditions, while soil models simulate the often highly heterogeneous below-ground environment that plants need to extract nutrients and water from. Crop models such as APSIM and CropSyst aim to integrate information about the weather, soil, crop management and the crop properties into yield predictions, in order to help improve management and breeding practices [83, 189].

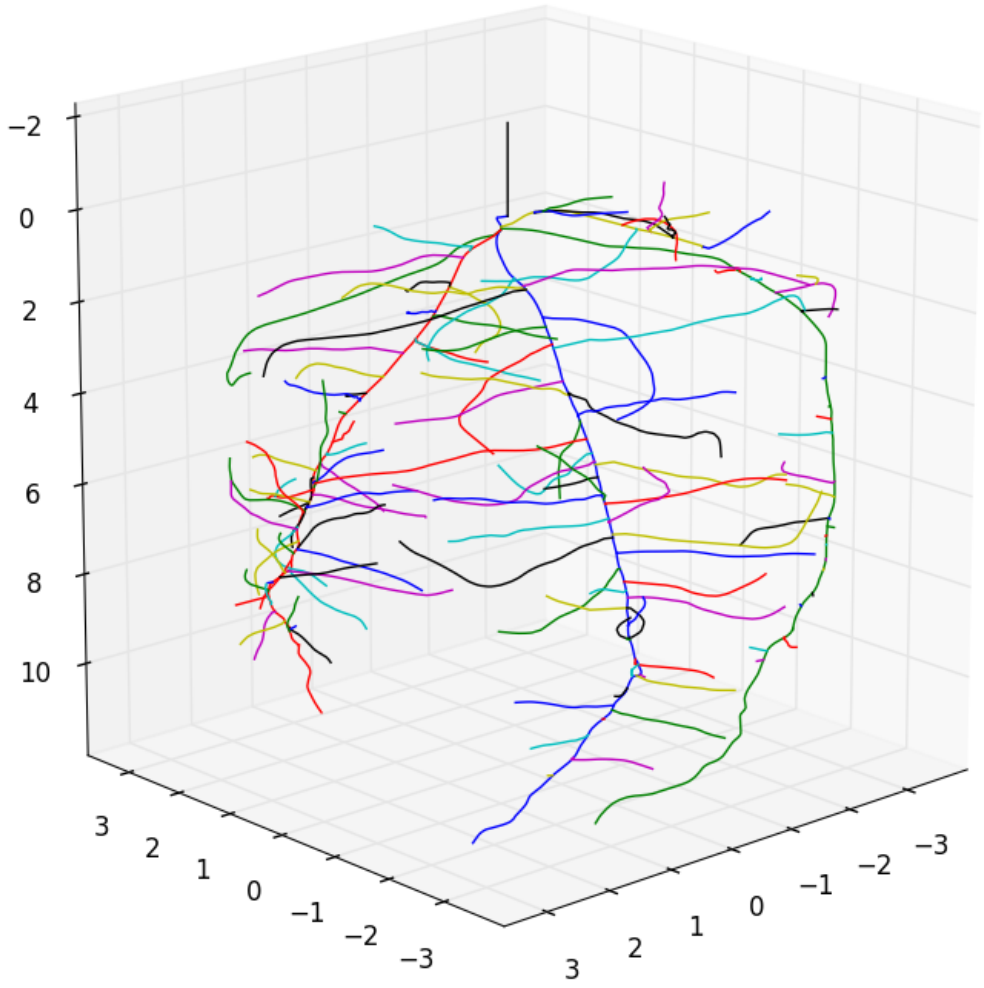


Figure 2.3: A wheat root system imaged at the Hounsfield CT facility as part of research conducted at University of Nottingham's Centre for Plant Integrative Biology (CPIB). The distance on the axes are in centimeters.

These models do not simulate individual roots in detail, instead modelling fields at timescales up to several years.

At the most simplified end of the scale are models representing roots as density distributions in the soil [70, 100]. These models are relatively easy to parametrise and provide a good simple reprentation of root systems. The simplified description means that the model model outcomes depend only on aggregate state variables and any heterogeneity within the soil is averaged out over a representative volume. The relatively new framework of continuous models provides a more detailed description of roots in soil without explicitly representing individual roots [19, 51]. These models represent roots using root length density and root branch-

ing density, whose behaviour is governed by differential equations, much like one simulates water flow through the soil. Such models can simulate soil heterogeneity at a smaller scale than root length density distributions, though the assumptions underlying them are not valid for scales below a certain representative volume, while still requiring relatively few parameters and being analytically and computationally tractable. The most detailed models contain explicit representations of roots, these are called root system architectural (RSA) models. These models typically require many parameters and are computationally expensive since each root or individual root segment is explicitly represented.

The earliest computer simulation model that explicitly represented individual roots contained a two-dimensional representation of roots and was able to calculate root length density in different soil layers based on root elongation and lateral branching rules and included the option to include root growth responses to fertilizer [110]. The first three dimensional RSA model represented tree root systems and considered the effect of wind on tree development [42]. Later models used similar growth rules to generate root systems and calculating outputs such as root length density [43, 145, 181]. Another model simulated wheat with root growth rates depending on temperature [152]. These early models were useful for providing a way to generate estimates of root system properties such as total root length and root length density distributions in the soil from simple branching and growth rules and allowed us to visualise root systems in a way not seen before. One RSA model was used to model soil exploitation efficiency of different root architectures using depletion zones around roots and their overlap [62].

The inclusion of biological mechanisms regulating the growth of roots in RSA models were a major advance. These models, termed functional-structural plant models (FSPM) combine a three-dimensional explicit representation of plant geometry with models for physiological plant functions [72, 203]. These models are based on the paradigm that plants respond to their environment in both adjusting physiological functions (e.g. photosynthesis, respiration) as well as structure (e.g. root elongation, branching rates) and that the structure determines in what environments plant organs operate while the functioning of plant organs has implications for the growth and thus the plant structure. It provides a modelling

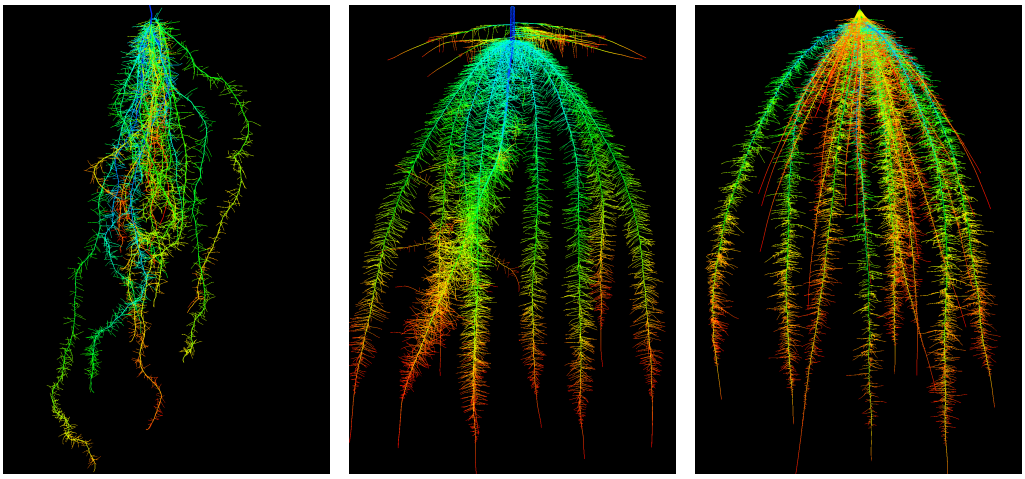
framework that allows for local as well as global interactions between plant organs. Similarly, the coupling of RSA models to models simulating the flow of water and nutrients in soil opened up new avenues of research and allowed more questions to be addressed.

The model presented in [38] simulated root growth based on soil temperature and impedance while also simulating water flow in the soil, root water uptake and carbon assimilation and allocation. The first version of SimRoot, the predecessor of OpenSimRoot, also included carbon allocation in the form of respiration, exudation and biomass [119]. Over time, RSA models incorporated more and more features, such as root plasticity and growth responses to the environment [33, 46, 78, 211], more detailed water and nutrient flows in the soil [179], uptake of water and nutrients and growth limitations under deficencies [47, 91, 102, 159, 176, 211], carbohydrate allocation models [139, 205, 211], rhizosphere processes [46, 155, 177, 211], root anchorage [50] and root traits such as root cortical aerenchyma [159]. For an overview of recent developments see [48] and [157].

2.9 OpenSimRoot

OpenSimRoot implements the simulation of models that capture the geometry, growth and nutrient uptake of root systems [119, 156]. Multiple plants can be simulated and because a simulated plant is specified through parameterized root classes OpenSimRoot can simulate many different root systems such as maize, bean, barley and rice [155, 154, 175, 74]. See figure 2.4 for visualisations of different root systems simulated using OpenSimRoot. OpenSimRoot predictions have been verified in the field. OpenSimRoot predicted that root cortical aerenchyma (RCA) improve growth under suboptimal nitrogen (N), phosphorus (P) and potassium (K) supply [159], which has been verified in field trials in South Africa and the USA [66, 174]. The advantage of RCA extends to drought-stressed plants as well [36]. OpenSimRoot also allows us to evaluate why traits have utility by looking at individual effects associated with those traits, which is not possible in real plants. For example, it was found that nutrient reallocation due to root cortical senescence (RCS) has a greater effect on plant growth than the associated reduction in respiration or nutrient uptake in barley under low N,

P and K availability [175]. OpenSimRoot is a useful tool to make predictions about the utility of phenes in specific environments, can provide more insight in exactly why phenes are beneficial or detrimental and allows us to study processes and environments which are difficult or impossible to study in the field. In this thesis, we will see the utility of OpenSimRoot in these three contexts.



(a) Barley simulated for 80 days.

(b) Bean simulated for 42 days.

(c) Maize simulated for 42 days.

Figure 2.4: Barley, bean and maize root systems as simulated by OpenSimRoot. The different colours indicate the root segment age, with blue colours being the oldest roots and red the youngest. The root systems are all 150 centimeters tall.

New functionality can be added to OpenSimRoot with relative ease because of the flexible structure of the code, which means that different submodels communicate through a common application user interface (API) and submodels can be added without needing knowledge of the internal coding of existing submodels. While there are some dependencies between modules, the user is mostly free to choose which are included in the simulation by specification in the input files.

In OpenSimRoot each root is simulated as a number of vertices connected by edges. The tip of each root is simulated by a vertex with time-dependent coordinates called the growthpoint. The speed of the growthpoint is defined by the base growth rate specified in the XML input file and possibly modifiers related to nutrient and carbon constraints and local soil conditions. The direction in which the growth point moves is determined according to rules relating to gravitropism, the emergence angle of roots and a stochastic contribution. The non-growthpoint vertices of a root, representing root segments, have fixed locations and are placed in

the path of the growthpoint as the root grows. The root length is the distance the growthpoint travelled, not the sum of the distances between root vertices. This is because these two quantities can differ slightly if the path the growthpoint takes is winding on length scales smaller than the distance between root vertices (like how the length of a coast line depends on the size of the ruler used to measure it).

New roots are created according to branching rules which specify the distance or time between subsequent branchings. Branches emerge from what are called xylem poles and the specified number of xylem poles determines the radial angles at which new branches can emerge. The XML input file specifies both the axial branching angle as well as the types of roots that can branch from a certain root class. Each root class has their own parameters, such as growth rates, branching rates, etc.

OpenSimRoot contains a simple, abstract canopy model in which the shoot is represented by a number of variables such as leaf area, shoot biomass, photosynthesis rate. A simple photosynthesis model determines the rate of carbon production based on the leaf area and the carbon fixation rate. The carbon requirements are based on the growth and respiration rates of the roots, costs associated to root exudates and nitrate uptake and the requirements of the shoot. The carbon needed for respiration, exudation, nitrogen fixation and nutrient uptake are first substracted from the total, with the rest being available for growth. If the remaining carbon is greater than the amount required for potential growth, leftover carbon is stored in a labile pool for later use. In the case of carbon availability being lower than potential growth rates require, growth rates decline, with shoot growth being prioritised, leading to a decrease in root-shoot ratio. Likewise, major root axes are given priority over laterals in carbon allocation to roots.

Water dynamics in OpenSimRoot are simulated with three models. One is a simplified implementation of the SWMS model in C++ which simulates water transport through the soil by solving the Richards' equation using the finite element method in combination with a finite differences method [179]. The Richards' equation is:

$$\frac{\partial \theta}{\partial t} = \nabla \left[K(\theta) \nabla (h(\theta) + z) \right] - S. \tag{2.9.1}$$

Here θ is the volumetric water content, t is time, $K(\theta)$ is the hydraulic conductivity tensor, $h(\theta)$ is the matrix head, z is the elevation above some reference point and S is a sink term that represents the water uptake by roots. Evapotranspiration, which is a term that includes the evaporation of water from the soil and transpiration by the plants, is simulated by the Penman-Monteith equation [3, 133, 134, 147].

The transport of water through the xylem is simulated by the hydraulic network model [4, 45]. The model assumes steady state flow, meaning that the capacity of the roots to hold water is negligible compared to the amount of water transpired, and assumes that the influence of solutes on the flow is negligible. The flow of water into the roots $J_r(z)$ and the flow inside the root, $J_h(z)$ are modelled as

$$J_r(z) = L_r(z)S(z)\left[\psi_s(z) - \psi_r(z)\right], \tag{2.9.2}$$

$$J_h(z) = -K_h(z)\frac{\partial \psi_r(z)}{\partial z}, \tag{2.9.3}$$

where z is the direction along the root, L_r is the radial hydraulic conductivity, S the root surface area, ψ_s the soil water potential, ψ_r the root water potential and K_h the axial hydraulic conductivity. By enforcing that the sum of fluxes at every root node is equal to zero (with the collar node, where the shoot meets the root system being the exception) we calculate the root water uptake for a given hydraulic potential at the collar by solving a matrix-vector equation. Using that this is ultimately a linear set of equations, OpenSimRoot calculates the required collar potential for a given potential transpiration rate.

The transport of nutrients in the soil is simulated with convection-diffusion equations for which there are currently two implementations in OpenSimRoot. One is the one-dimensional Barber-Cushman model that is used to simulate phosphorus uptake and to simulate depletion zones around root segments [89]. The second is an implementation of the solute model in SWMS3D that couples to the water transport model [179]. The uptake of mobile nutrients by the root system is modelled with Michaelis-Menten kinetics. The nutrient uptake rate of a root

segment, I, is equal to

$$I = \begin{cases} \frac{I_{max}(C-C_{min})}{K_m+C-C_{min}} & \text{if} \quad C \geq C_{min}, \\ 0 & \text{if} \quad C < C_{min}. \end{cases} \tag{2.9.4}$$

Here I_{max} is the maximal uptake rate of the root segment, C is the nutrient concentration at the root surface, C_{min} is the minimal nutrient concentration at which the root segment can take up nutrients and K_m is the concentration at which $I = \frac{I_{max}}{2}$. By comparing the total nutrient uptake with specified minimum and optimal nutrient concentrations for each plant organ, OpenSimRoot calculates a stress factor for each nutrient. This stress factor impacts root elongation rates, branching rates, photosynthesis rates, respiration rates and leaf growth rates through transfer functions specified in the input files. By scaling back shoot growth first under nutrient stress, the increase in root-shoot ratio under nutrient stress observed in many species becomes an emergent feature of the model. Nitrogen fixation such as happens in legumes can also be modelled. Mineralisation is modelled by the Yang-Janssen model [215].

Various root morphological features such as root hairs, root cortical senescence and root cortical aerenchyma can be simulated in OpenSimRoot. These impact water and nutrient uptake, root maintenance costs and the carbon and nutrient content of roots. Roots can also respond to local nutrient concentrations by altering their elongation and branching rates or adjusting their gravitropic response through transfer functions specified in the input file.

OpenSimRoot is written in C++, an object-oriented programming language. The object-oriented programming paradigm is well suited to the simulation of root systems; root segments are represented by instantiations of the same object, added during growth. In OpenSimRoot, every root segment object will be coupled to various objects encoding relevant state variables corresponding to that root segment, such as its diameter, volume, dry weight and nutrient uptake rate. So every minimodel in OpenSimRoot, which represents a single state variable, corresponds to a C++ class which inherits from a common base class called SimulaBase. This inheritance provides each minimodel with a common set of methods which allow the engine to handle any new minimodel and are used to pass information be-

tween minimodels. Helper functions facilitate the correct behaviour of models within the simulation and facilitate the passing around of information. These helper functions provide caching to avoid repeating calculations if not necessary and allow for interpolation of values, as well as integration of values through integration functions. Through this separation of minimodels and certain functionality, developers don't need detailed knowledge of the OpenSimRoot engine to add new models; knowing how to request the values of relevant variables and the appropriate helper functions to associate with each minimodel is all that is required.

Internal dependencies between models and the order in which calculations are executed are determined at runtime. OpenSimRoot simulations are driven by the outputs which are specified and models are only activated when values are, possibly indirectly, requested by the output module. When a minimodel depends on another minimodel, it requests information through the application user interface (API), which prompts the other minimodel to make the necessary calculations and return the requested value. This structure allows for a complete specification of the parts included in the simulation through the XML input files without requiring any modification to the source code.

OpenSimRoot allows for integrating values forward in time using a variety of different integration functions such as forward Euler and Heuns. The default integration function is fourth order Runge-Kutta (RK4). For an initial value problem of the form

$$\frac{dy}{dt} = f(t, y), \quad y(t_0) = y_0, \tag{2.9.5}$$

this method works as follows. For a step of size $h > 0$,

$$y_{n+1} = y_n + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4), \tag{2.9.6}$$

$$t_{n+1} = t_n + h. \tag{2.9.7}$$

Here y_n is the approximation of $y(t_n)$ by the RK4 method. k_1, k_2, k_3, k_4 are

defined as

$$k_1 = f(t_n, y_n), \tag{2.9.8}$$

$$k_2 = f\left(t_n + \frac{h}{2}, y_n + h\frac{k_1}{2}\right), \tag{2.9.9}$$

$$k_3 = f\left(t_n + \frac{h}{2}, y_n + h\frac{k_2}{2}\right), \tag{2.9.10}$$

$$k_4 = f(t_n + h, y_n + hk_3). \tag{2.9.11}$$

Since the RK4 method is of fourth order, the error is on the order of $O(h^4)$. So while each RK4 step takes more computation than, say, a step of forward Euler, because it is more accurate we can take bigger timesteps. For most models, the default timestep (which can be specified in the input file) is sufficient but OpenSimRoot allows for different timesteps between models, which is relevant if a certain precision is required or if one wants to simulate processes at different timescales. If there is a mismatch between timesteps, then values are interpolated if needed. Quantities being integrated forward in time can indirectly depend on themselves, for example the leaf elongation rate depends on carbon availability, which depends on photosynthesis rate which depends on leaf area and hence leaf elongation rate. To integrate forward in time when this happens OpenSimRoot uses a predictor-corrector method. This means that integration is done once, the result is saved as a prediction and once all dependencies are resolved, the integration is redone for all predicted values to arrive at the result. The OpenSimRoot engine tracks dependencies between different minimodels and uses this to determine if values are final or still part of a prediction (for example if a value is calculated as intermediate value in an integration step).

OpenSimRoot is available under the `https://www.gnu.org/licenses/gpl-3.0.en.html`GPLv3 License. This is an open-source copyleft license. This means that anyone can freely download and modify OpenSimRoot as they wish, and any code which includes the OpenSimRoot source code has to be published under the same license. This ensures the model will remain accessible to scientists. The source code can be downloaded from `https://gitlab.com/rootmodels/OpenSimRoot`, where issues can be created as well. OpenSimRoot is under active development and the code is updated whenever improvements are made or bugs

are addressed. Any new features will be implemented in a backwards-compatible fashion if possible. This means that old input files will produce the same result with the newest version of the code as with the version that was originally used. Note that this is not always possible with major updates to the OpenSimRoot engine.

For more details on OSR, see [119, 118, 156] or appendix D, which also provides a quick start guide for potential users.

Chapter 3

Root Loss and Nutrient Uptake

Here follows an article which was sent in to Plant Physiology. The article is currently in revision after a submission which lead to useful feedback. The text is reproduced here in slightly edited form. The non-supplemental figures have been put in the text and the bibliography was integrated with that of the entire thesis. The supplemental figures can be found in Appendix A.

3.1 Abstract

Despite the widespread prevalence of root loss in plants, its effects on crop productivity are not well understood. While root loss reduces the capacity of plants to take up water and nutrients from the soil, it may provide benefits by decreasing the resources required to maintain the root system. Using the open-source, functional-structural root system simulation model OpenSimRoot, a range of root phenotypes were simulated in different soils and root loss scenarios for barley, common bean and maize. The simulations predicted root loss is detrimental for phosphorus uptake in all tested scenarios and reduces nitrogen uptake in most tested scenarios. Loss of main root axes reduced predicted fitness for all phenotypes in all species and soils, whereas lateral root loss had smaller or no detrimental effects. In low-nitrogen, high-phosphorus soils, some maize phenotypes even showed a small increase in shoot biomass after lateral root loss. The best predictor of shoot biomass reduction due to root loss was the maximum cumulative root loss a plant endured, as fraction of total root length, rather than

the absolute or relative amount of root loss. Bean was much more resilient to axial root loss than barley and maize and resilience to axial root loss correlated with the amount of axial roots formed early in development. We conclude that root loss is an important factor in the evolution of root architectural phenotypes, the effects of which depends on species, phenotype, nutrient availability and the type and intensity of root loss.

3.2 Introduction

Roots are vital plant organs that forage for nutrients and water, provide anchorage, and provide storage in selected species. Because of inadequate use of soil fertility inputs, soil degradation and adverse effects of global climate change, crops in many locations face challenges accessing adequate nutrients. Added to this, plants are in constant competition for the resources that are available, both below and above ground and are constantly under threat from herbivory, root rots, disease and nutrient deficiency, as well as environmental stresses such as heat, cold and drought. Their root systems are not exempt from these threats and root loss is prevalent in many species. A meta-analysis of 85 studies into the effects of 36 species of root feeding insect herbivores on 75 plant species found that belowground herbivory led to an average reduction in root biomass of 36.3% [222]. Understanding root growth and functioning under such adverse conditions is vital for understanding a fundamental dimension of plant fitness, and has agricultural relevance in guiding the development of the more resilient, sustainable crops urgently needed in global agriculture [115]. Understanding the effects of root loss in this context is important because root phenotypes that are associated with high yields in controlled or high-input environments with minimal root loss might perform poorly in conditions where root loss is prevalent.

There is evidence root loss is not programmed, in contrast to leaves [61]. Root length does generally decrease during and after flowering, this could be because plants reduce investment in new roots during this stage of their development [55]. Several external factors affect the prevalence of root loss. Drought increases root turnover and topsoil drying leads to rapid root dieback [75]. Although reducing soil temperature appears to have no clear effect on root longevity of trembling

aspen [96], increasing soil temperature has been found to increase root mortality in grasses [63], white clover [207] and sugar maple [77]. The mean annual temperature was the most important variable explaining fine-root turnover in the global data set on root turnover of Gill and Jackson [71], and it suggests a mean annual temperature increase of 10°C leads to a 40-90% decrease in root lifespan. There are many variables that vary with temperature and this makes it difficult to ascertain if the effect on root longevity is a direct consequence of temperature or something else such as pathogen activity or soil quality. It should also be noted that some studies found increasing temperature had no clear effect [87, 183]. Similarly, the availability of soil nutrients appears to influence root lifespan but there are conflicting results. Some studies found that high nutrient availability coincides with short root lifespan [123, 161, 162, 185], while others found the opposite [1, 26, 160]. Mycorrhizal associations seem to protect roots from a variety of factors, enhancing their longevity [56, 67, 137]. There are many organisms in the soil, some of which feed on roots [209], explaining why applying pesticides increases root lifespan considerably [208]. Hence root longevity depends on the plant, herbivores and pathogens.

Even if "programmed" root loss (referred to as root senescence) does occur, many of the resources invested in a root are lost when the organ senesces, as are its capacity for water and nutrient uptake and that of any roots that subtend from it. Losing roots of certain types may also affect plant anchorage and stability. In this paper we will explore the effects of lost resources and reductions in uptake capacity, due to root loss, on plant fitness.

Nevertheless, root loss may also have benefits. The amount of resources invested in the production and maintenance of root systems is considerable, as evidenced by the fact that the dry weight of root systems can be larger than the dry weight of the shoots [32, 64, 105]. Out of all the carbohydrates produced by photosynthesis, more than a fifth can be spent on root respiration [8, 150], under phosphorus stress this can increase to more than 40% [117, 138], while more than 15% of the carbohydrates can be spent on root exudates [81]. Roots that are lost do not have to be maintained and it is in this way that root loss can lead to a very large reduction in resource expenditure. What further illustrates this is that the car-

bon cost of respiration necessary to maintain a root can exceed the carbon cost of growing the root in as little as 20 days [55]. Thus, it is reasonable to suggest that root loss may be beneficial in some situations. It has been suggested that parsimonious, or sparse, root systems perform better in drought conditions because they can reach deeper soil strata by virtue of being less costly to maintain [112, 114]. Since nitrogen leaches down into the soil, parsimonious root phenotypes, such as root systems subjected to moderate to high rates of root loss, could be beneficial in conditions of suboptimal nitrogen availability [115].

There have been suggestions that root loss can be beneficial in low-nutrient environments. Low soil phosphorus concentrations were observed to lead to increased root turnover [185]. A previous root-system model predicted that root turnover increases the explored soil volume and therefore phosphorus and potassium uptake, although this model did not assign any cost to root turnover [184, 185, 187, 186].

Root phenotypes can differ greatly among plant species. Dicot root systems, such as common bean, start from a primary root which develops into a taproot which develops through lateral root formation, and in some cases basal roots as in common bean. In contrast, the lack of secondary growth in monocots makes it necessary to continually produce nodal roots of increasing diameter from shoot tissue. Many grass species, including principal cereal crops like wheat, rice, and barley, form tillers, which grow root systems themselves in turn. As a result of crop breeding, modern maize lines rarely tiller. There are also interspecific and intraspecific differences in lateral branching rates, the number of lateral root orders and the number of adventitious roots that plants grow. It has been hypothesised that these differences in root architectural phenotypes, and the resulting differences in root system topology, lead to differences in susceptibility to root loss among species and phenotypes [113].

One of the reasons why experimental research on root loss is limited in the literature is that it is difficult to study in the field. Studying roots in their natural environment is much harder than studying shoots, due to the difficulty of visualising roots in situ within the soil, and studying dynamic processes such as

root loss is particularly challenging. Invasive methods such as rhizotrons alter the local soil environment and allow one to study only a small part of the root system. To detect root loss one needs to identify roots across multiple images. But it can also be difficult to determine if a given root has been lost or not, since roots and soil might shift around and determining the status of a root from visual inspection alone can prove difficult. Soil coring or excavating root systems only provide snapshots of root system development and even if recently deceased roots can be identified it is difficult to determine overall root loss rates.

The significant challenges associated with experimental studies of root loss make modelling very useful. Not only does modelling permit precise control over root loss rates, which is all but impossible in field conditions, but it also allows access to information that would be very hard to obtain in field experiments, such as root development over time, nutrient uptake rates as well as the complete structure of the root system. Simulations also allow us to study a much larger array of phenotypes and environmental scenarios than would be possible in field experiments, without any factors such as the weather being out of our control. Currently a number of different root system architecture models exist that simulate root growth and functioning [43, 91, 103, 146, 156, 211]. We have used OpenSimRoot [156], the open-source successor of SimRoot [119, 118], because it allows us to simulate resource acquisition and allocation as well as the effects of shortages. This makes it an ideal simulation model to study the impact of root loss. (Open)SimRoot has been used to simulate barley, bean, lupin, maize and squash in a variety of settings [34, 68, 158, 155, 154, 164, 175, 190, 205]. OpenSimRoot simulates the geometry, growth and nutrient uptake of root systems, as well as water and nutrient flows in the soil [119, 156]. Because OpenSimRoot simulates the development of root systems through the application of growth and branching rules for each root class, it can simulate a wide variety of different root architectures. New functionality can be added with relative ease because of the modular structure. While there are some dependencies between modules, users are free to choose which modules are included in the simulation.

With this study, we are extending functional-structural plant models by adding a root loss module to OpenSimRoot. We present results from root system sim-

ulations of common bean (a dicot), barley (a tillering grass), and maize (a non-tillering grass) subjected to various levels and types of root loss in different soil environments in order to study the effects of root loss on plant productivity in different environments.

3.3 Materials and methods

3.3.1 Root Loss Module

Root loss is simulated by deactivation of root segments. The time of root loss can be determined in a number of different ways, see below, and once this time has passed, the root segments in question are considered lost. These root segments do not take up any more nutrients or water, do not count towards total root length, root mass, etc, do not respire or need any other resources. Any root subtending from a deactivated root will be deactivated as well. If a root segment is lost, the deactivation is propagated downwards towards the apex. Needless to say, deactivated apices stop moving.

The root-loss module keeps track of the root length that has been lost during the simulation and the amount of carbon that has been lost, based on the carbon content of the lost roots. It also simulates the loss of nutrients such as N, P and K in the tissues of the lost roots by subtracting this from the nutrient pool. The amount of nutrients lost is calculated by assuming that nutrients are distributed homogeneously in the plant tissue, weighed by the minimal and optimal nutrient contents in each tissue, depending on nutrient stress levels, as quantified by the nutrient stress factor. The stress factor S is calculated as follows:

$$S = \begin{cases} 0 & \text{if } U \leq P_m \\ \frac{U - P_m}{P_o - P_m} & \text{if } P_m \leq U \leq P_o \\ 1 & \text{if } U \geq P_o \end{cases} \tag{3.3.1}$$

Here S is the stress factor for the nutrient under consideration, U the amount of that nutrient currently in the plant (initial seed content plus uptake up to now minus nutrients lost up to now) in µmol, P_m the minimal nutrient content of the plant in µmol and P_o the optimal nutrient content of the plant in µmol. The

amount of nutrients lost by the plant, L_σ when a segment σ is lost is equal to:

$$L_\sigma = \begin{cases} (1-S)\sigma_m + S\sigma_o & \text{if } 0 \leq S \leq 1 \\ \frac{U \cdot \sigma_o}{P_o} & \text{if } S = 1 \end{cases} \tag{3.3.2}$$

Here $L_\sigma(S)$ is the amount of nutrients lost when segment σ is lost in µmol, S is the stress factor as defined above, σ_m is the minimal nutrient content of segment σ in µmol and σ_o is the optimal nutrient content of segment σ in µmol. Partial nutrient remobilisation can be simulated by adding a parameter set to the appropriate remobilisation value R. Then the amount of nutrients lost per segment, L'_σ will be calculated according to the expression:

$$L'_\sigma = (L_\sigma - \sigma_m) \cdot (1 - R) + \sigma_m \tag{3.3.3}$$

Here L_σ is defined as above. With maximum remobilisation ($R = 1$), the plant will lose σ_m per segment, the minimal nutrient content needed for the tissue to function normally.

The time of root loss can be determined for each segment individually or for entire roots at once, and each root class can be assigned different root loss probabilities or lifetimes by specifying this in the input files. The time a root (segment) is lost is determined based on a distribution of root lifetimes, such as a uniform or normal distribution, or a daily probability of root loss. If a daily probability for root loss is chosen, the probability can be modified based on depth in the soil. Because of the modular structure of OpenSimRoot it is straightforward for anyone familiar with C++ to add new plugins that determine the probability of root loss based on local soil conditions, the water or nutrient status of the plant or root characteristics such as root diameter or age.

3.3.2 Simulated Scenarios

Three crop species, barley, bean and maize, were simulated in a variety of scenarios. The parameters used in our simulations either come from previous (Open)SimRoot publications or were estimated from the literature. Parameters used in previous publications mostly come from field or greenhouse experiments and results from many of these publications have been verified in experiments. As summarised in

Table 3.1, for each species, we had root cortical senescence (RCS)(barley) or root cortical aerenchyma (RCA)(bean and maize) either present or absent and varied the lateral root branching density (LRBD) and the axial root number, each with 3 different values.

	Barley	Bean	Maize
RCA/RCS	No RCS/RCS	No RCA/RCA	No RCA/RCA
LRBD	1.6/2.5/5 $\frac{\text{branches}}{\text{cm}}$	2/4/6 $\frac{\text{branches}}{\text{cm}}$	2/5/20 $\frac{\text{branches}}{\text{cm}}$
Axial root number	2/3/4 tillers	4/12/16 basal roots 10/25/40 hypocotyl born roots	11/18/36 nodal roots 12/34/34 brace roots

Table 3.1: An overview of the three phenes that were varied for all three species and the values used in the simulations. A factorial design was used, which means that every combination of values for these three phenes was simulated.

All of these phenotypes, 18 for each species, were simulated in a variety of environments. They were placed in 4 different nutrient environments, high and low availability for both nitrogen and phosphorus. Each phenotype was subjected to three different types of root loss: lateral root loss, axial root loss and a combination of both types of root loss. Root loss was simulated by assigning to each root of the relevant root type a daily probability of that root being lost. All three of these types of root loss were simulated at three different levels of severity, with roots having a 1%, 2.5% or 5% daily probability of being lost. A root loss rate an order of magnitude lower than 1% per day was deemed unlikely to affect plant development significantly over the 40 or 80 simulated days while a root loss rate of 10% per day was expected to be too detrimental for plants to develop under. With these numbers we hope to have a range of values covering most scenarios where plant growth is reduced but the plant is able to survive. A control simulation without any root loss was also done. Each simulation was repeated 5 times with different seeds of the random number generator. This is because root loss, root branching and root growth direction all include a stochastic component.

Instructions on how to recreate the XML input files and the OpenSimRoot executable used to generate the data for this paper can be found at:
https://gitlab.com/rootmodels/OpenSimRootPapers/-/tree/

master/ErnstDSchafer-2020-RootLossAndNutrientUptake.
The version of the code used for the simulation in this paper corresponds to the code with git commit hash 5863e3e6f14927d21e4dcd903d75c4d4edb8d111 in the OpenSimRoot repository. It can be downloaded from
https://gitlab.com/rootmodels/OpenSimRoot/-/tree/5863e3e6f14927d21e4dcd903d75c4d4edb8d111.

OpenSimRoot is currently only able to simulate vegetative growth and species are only parametrised for a certain number of days. Because of this, we cannot simulate flowering and grain filling and hence yield. Even though the empirical relationship between shoot biomass and yield, called harvest index, varies depending on a number of factors including environment, field management and plant water status [194], shoot biomass is the best proxy we have available.

3.4 Results

Root loss impact on root system architecture is stage and order dependent

The root systems of barley, bean and maize are very different, in size as well as distribution of roots (Figure 3.1). Applying root loss at different intensities led to large differences in their root systems at 40 days (Figure 3.2). The root systems of plants growing under low intensity root loss conditions still looked very similar to those growing without any root loss, while at higher intensities very little of the root system remained. The differences, arising from stochasticity in the simulation, between root systems of the same phenotype subjected to the same root loss intensity was quite large, depending on the amount of root loss occurring early in development (Figure 3.3). Plants that lost a lot of axial roots early were not able to recover and ended up with small root systems, while those losing axial roots later in development were able to still grow and maintain a sizeable root system.

Shoot biomass decreases under most types of root loss in most environments

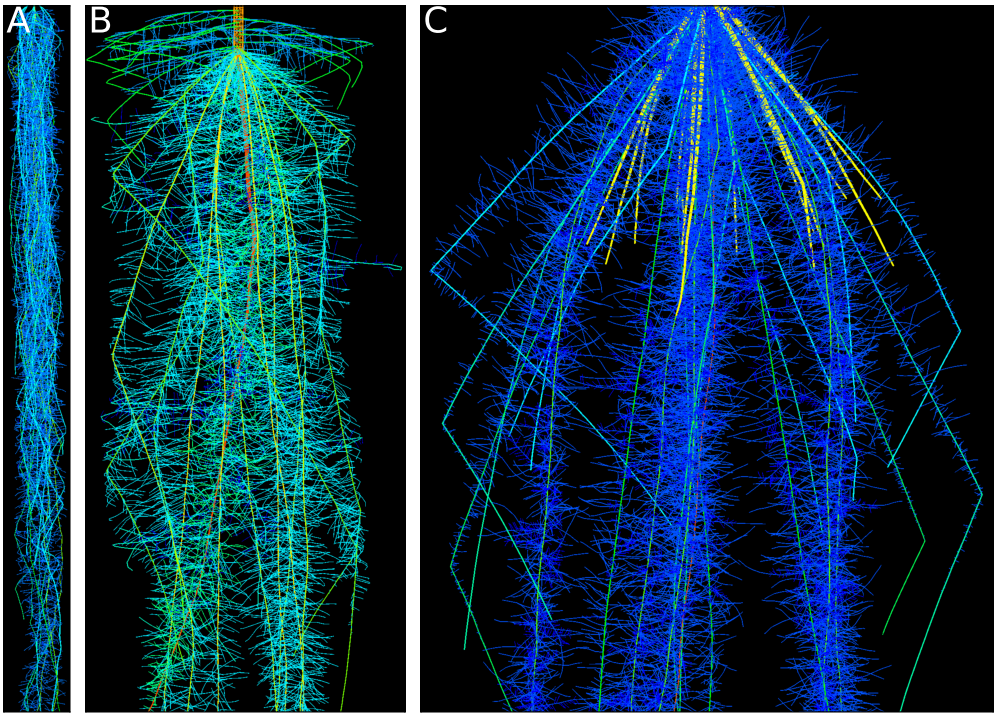


Figure 3.1: Root systems of (A) barley, (B) bean and (C) maize 40, 80 and 40 days after germination, respectively. The scales are the same and different colours indicate different root classes. For barley, the primary root is green, seminal roots are yellow, tiller roots are cyan and lateral roots are blue. For bean, the hypocotyl is light orange, the primary root is dark orange, seminal roots are yellow, hypocotyl borne roots are green, first order laterals are cyan and second order laterals are dark blue. For maize, the primary root is orange, seminal roots are green, nodal roots are cyan, basal roots are yellow, first order laterals are blue and second order laterals are dark blue. Soil column size is set to an area a plant typically has in the field. Roots are reflected when they reach the edges of the soil column to simulate the overlap between root systems of neighbouring plants in field conditions.

The sensitivity of simulated shoot dry weight to root loss depended on species, soil nitrogen and phosphorus availability and root loss type (Figures 3.4, 3.5, 3.6). Axial and general root loss led to large reductions in simulated shoot dry weight in all species and soils, with average decreases of 85% to 96% (barley), 37% to 56% (bean) and 80% to 89% (maize) compared with no root loss, depending on soil. In the low nitrogen, low phosphorus soil, lateral root loss reduced simulated shoot dry weight by 15-41% in barley and by 4-30% in bean compared with the scenario without root loss, depending on phenotype. In maize, the phenotypes with high lateral root branching density (LRBD) increased simulated shoot biomass under lateral root loss by 7-14% compared with the case without

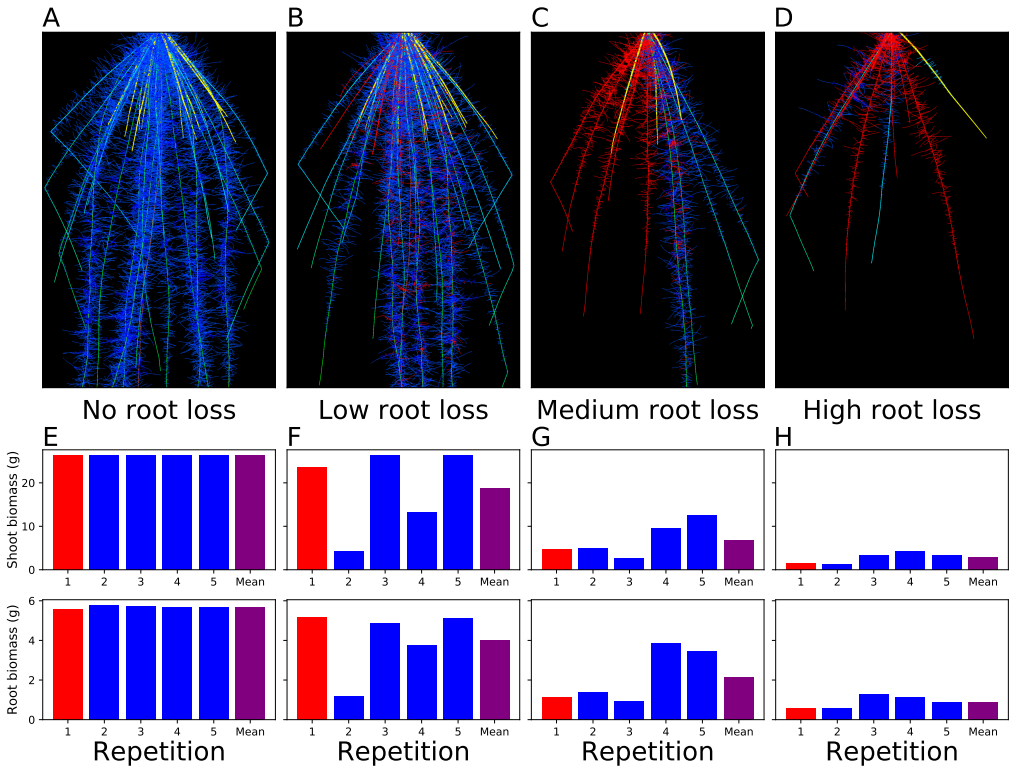


Figure 3.2: (A) A maize root system after 40 days in the high nitrogen, low phosphorus soil, without any root loss. The different colours indicate different root classes and red roots have been lost. Panels (B), (C) and (D) show similar root systems but grown under general root loss at low, medium and high intensity respectively. (E) Shoot and root dry weights in grams for all 5 repetitions with the same parameters as the simulation displayed in panel (A), with the red bar indicating the specific repetition shown in (A). The purple bar indicates the mean of these 5 repetitions. Panels (F), (G) and (H) are the same as panel (E) but corresponding to the root systems shown in (B), (C) and (D), respectively.

root loss, while the other phenotypes showed a decrease of 18-41%. In the high nitrogen, low phosphorus soil, lateral root loss reduced simulated shoot biomass for all species, with 26-57% for barley, 36-52% for bean and 3-41% for maize. In the low nitrogen, high phosphorus soil, changes in simulated shoot biomass were -11 to +8% (barley), -10 to +3% (bean) and -16 to +15% (maize) under lateral root loss, as compared to without root loss.

The results of the reference simulations, those without root loss, were largely in line with those of earlier studies. Root cortical senescence (RCS) in barley was associated with a large increase in shoot dry weight in all soils. Averaged over all other phenes in the low-nitrogen, low-phosphorus soil, the increase associated with RCS was almost 75%, in accord with the results from [175]. Root cortical aerenchyma (RCA) increased root dry weight in all soils with a few percent in

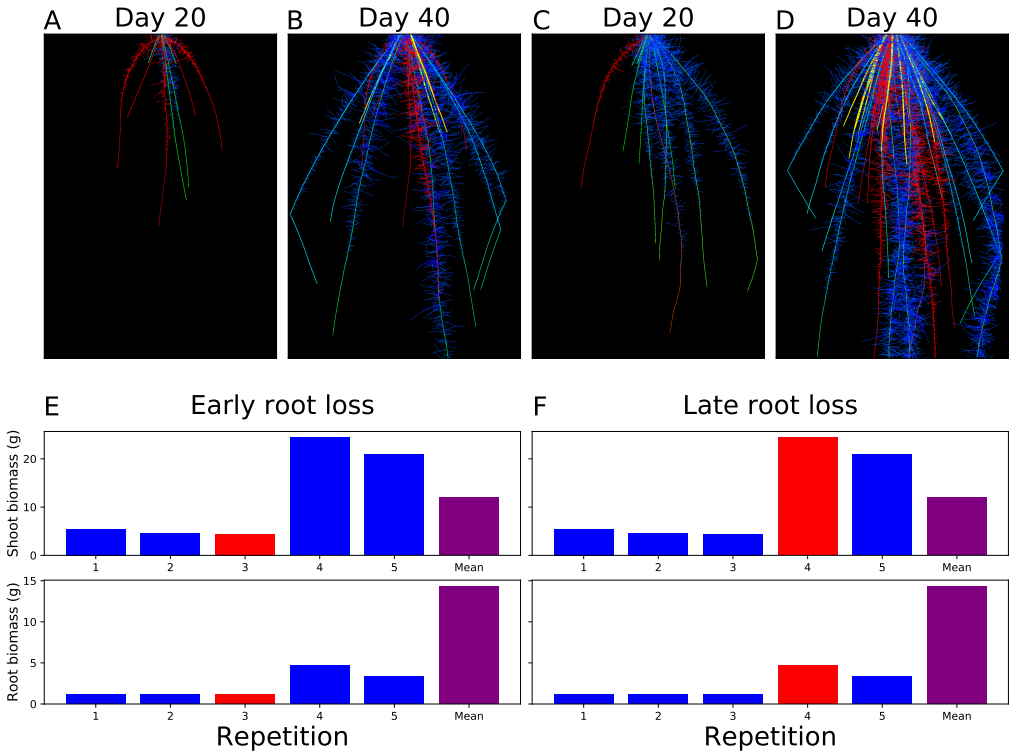


Figure 3.3: Panels (A) and (B) show the same maize root system subjected to axial root loss at 20 and 40 days, respectively. The different colours indicate different root classes and red roots have been lost. (E) Shoot and root dry weights for all 5 repetitions with the same parameters as the simulation displayed in panels (A) and (B), with the red bar indicating the specific repetition shown in panels (A) and (B). The purple bar indicates the mean of these 5 repetitions. Panels (C) and (D) show a different repetition from this set at 20 and 40 days, respectively. Panel (F) is the same as panel (E) but for the repetition shown in panels (C) and (D). The root system shown in panels (A) and (B) has lost a lot more roots early during development than the root system shown in panels (C) and (D), this let to large differences in root and shoot biomass at day 40, as shown in panels (E) and (F).

bean and up to 20% in maize, again averaged over all other phenes, smaller benefits than were observed in [158] and [159]. In the low-nitrogen, high-phosphorus soil, lower branching densities were associated with greater shoot dry weight in barley and especially in maize, where there was a factor 3 difference in averaged shoot dry weight between the low and high branching phenotypes, in line with previous in silico [154] and field [219] studies. In bean, the high branching phenotypes outperformed the low branching phenotypes by about 5%.on average. In the high-nitrogen, low-phosphorus soil, high branching densities increased shoot dry weight, averaged over all other phenes, by more than 100% in all three species, in line with previous in silico [154] and field [92] studies. Across all soils, low tiller numbers and low nodal root numbers increased shoot dry weight in barley and

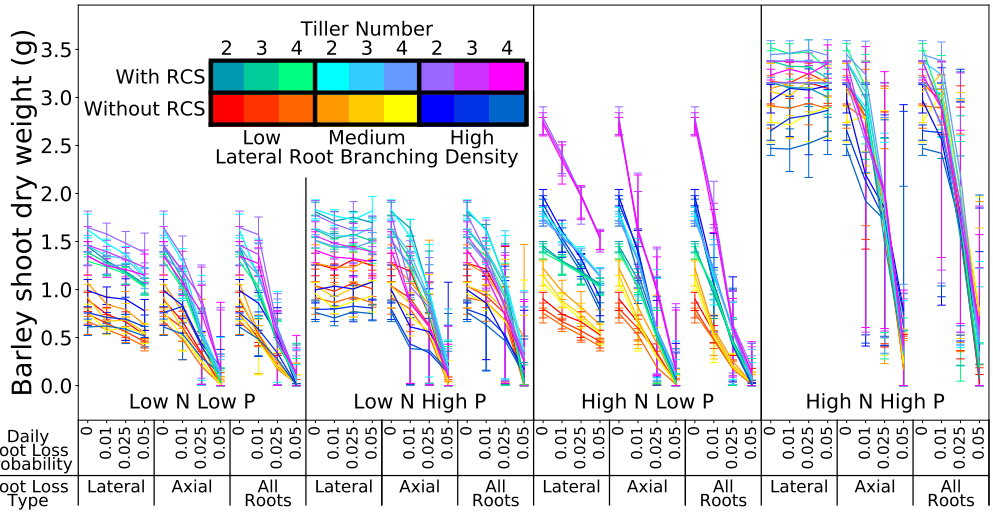


Figure 3.4: Mean shoot dry weights at 80 days of the 18 barley phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

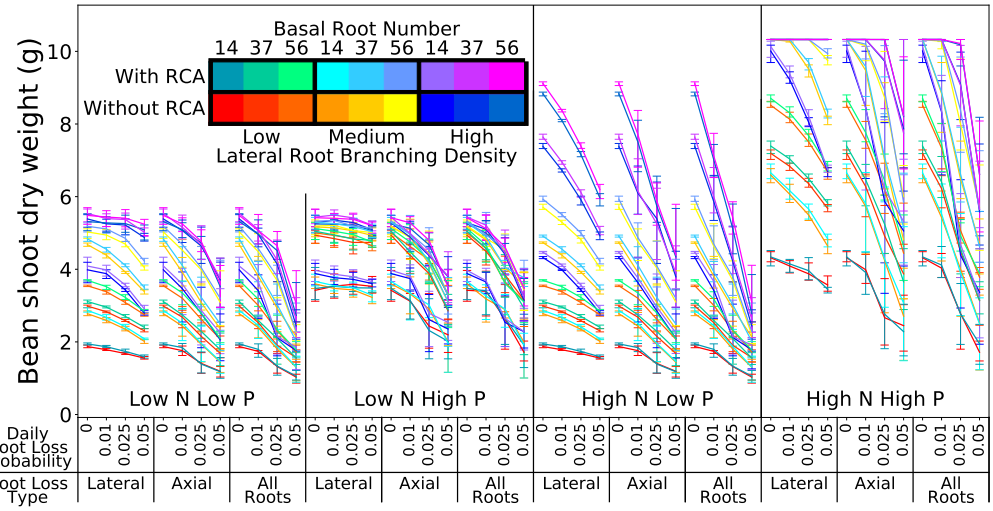


Figure 3.5: Mean shoot dry weights at 40 days of the 18 bean phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

maize respectively, while in bean, greater basal root numbers were associated with greater shoot dry weight.

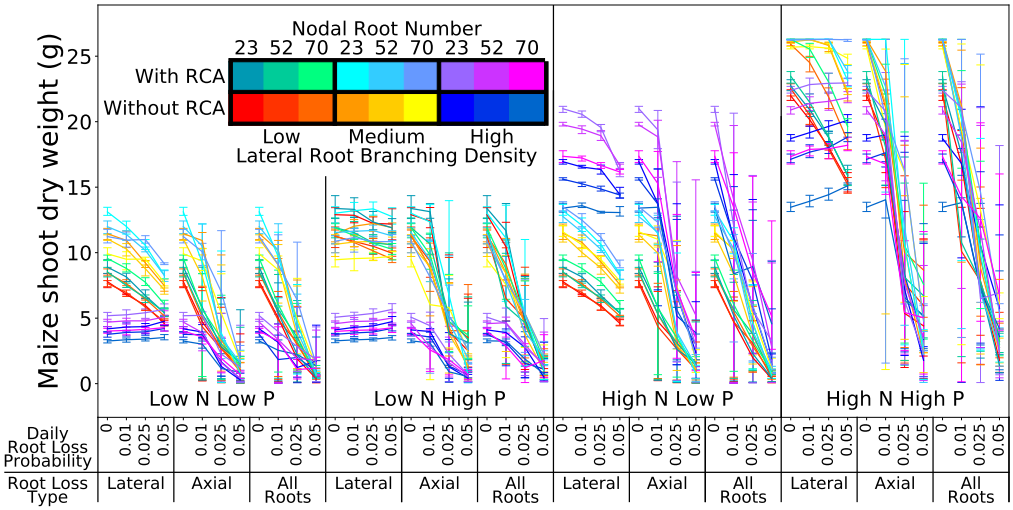


Figure 3.6: Mean shoot dry weights at 40 days of the 18 maize phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

The shoot dry weights of different phenotypes were mostly affected in the same way by root loss, with the increase or decrease depending largely on root loss type and intensity and soil conditions. The only exceptions were maize phenotypes with high LRBD, which increased shoot dry weight under high levels of lateral root loss in every soil except the high-nitrogen, low-phosphorus soil, contrary to the trend that other phenotypes followed (Figure 3.6). There was a strong negative correlation between the maximum root loss fraction (the maximum taken over the daily outputs in the simulations) and the change in shoot dry weight due to root loss in all three species (Figures 3.7, 3.8, 3.9). The greatest reductions in shoot dry weight as compared to the case without root loss occurred at high maximum root loss fractions, irrespective of nutrient availability or root loss type. The maximum root loss fractions were greater under axial and general root loss, greater than 0.9 for all three species, than under lateral root loss, where the largest values were 0.71 for barley, 0.50 for bean and 0.45 for maize. Averaged over all simulations with root loss, the maximum root loss fraction was 0.64 for barley, 0.42 for bean and 0.51 for maize.

Lateral root loss has little effect on nitrogen uptake, axial root loss leads to sharp decreases in nitrate uptake

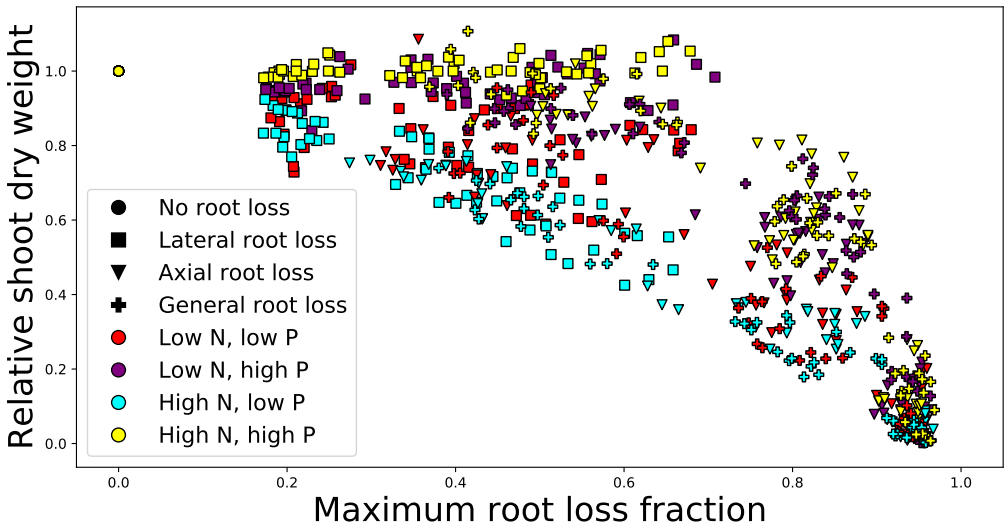


Figure 3.7: Shoot dry weight relative to the reference case without root loss versus the maximum root length lost, as fraction of total root length lost produced at that time, that was observed during the 80 days of simulation for barley. Each point represents the mean of 5 repetitions, the shape of the point indicates the type of root loss and the colour of the point indicates the soil nutrient availability.

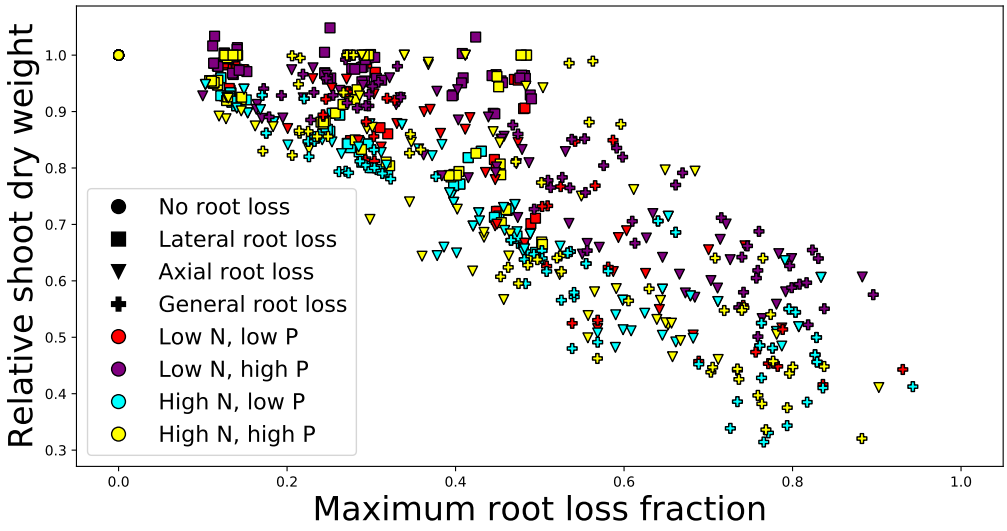


Figure 3.8: Shoot dry weight relative to the reference case without root loss versus the maximum root length lost, as fraction of total root length lost produced at that time, that was observed during the 40 days of simulation for bean. Each point represents the mean of 5 repetitions, the shape of the point indicates the type of root loss and the colour of the point indicates the soil nutrient availability.

In barley, lateral root loss had little effect on nitrogen uptake, but axial and general root loss led to a decrease in nitrogen uptake of up to 90% (Figure A.1). Differences in uptake rates between different phenotypes were small but high branching densities were associated with slightly higher nitrogen uptake and the

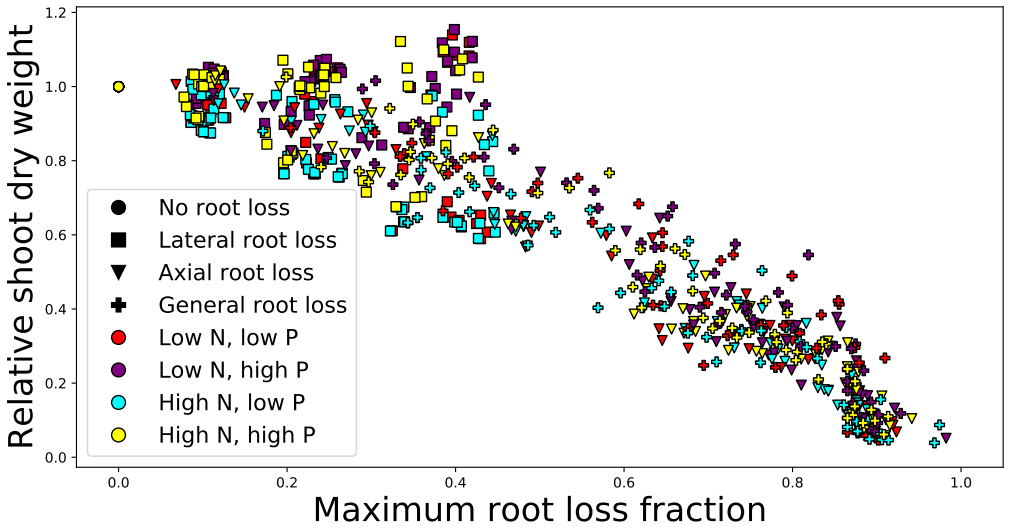


Figure 3.9: Shoot dry weight relative to the reference case without root loss versus the maximum root length lost, as fraction of total root length lost produced at that time, that was observed during the 40 days of simulation for maize. Each point represents the mean of 5 repetitions, the shape of the point indicates the type of root loss and the colour of the point indicates the soil nutrient availability.

presence of RCS reduced nitrogen uptake, regardless of the presence, type or intensity of root loss. In bean, lateral root loss was associated with a decrease in nitrogen uptake of up to 20% for most phenotypes, although in the low nitrogen soils some saw an increase in nitrogen uptake (Figure A.2). Axial and general root loss were associated with decreased nitrogen uptake of around 50% for most phenotypes, with some exceeding this. Bean phenotypes with medium or high basal root numbers had up to 30% greater nitrogen uptake than those with low basal root numbers and higher branching densities were associated with greater nitrogen uptake as well, irrespective of the presence of root loss. RCA had almost no influence on nitrogen uptake. In maize, lateral root loss led to an improvement in nitrogen uptake for the high branching phenotypes, (more than 30% for one phenotype), and a reduction or no effect for the other phenotypes (Figure A.3). Axial and general root loss led to large decreases in nitrogen uptake. Nitrogen uptake was strongly dependent on lateral root branching density: high branching density led to a 70% reduction of nitrogen uptake in the low nitrogen soils, as compared to low or medium branching densities, if no root loss was present but under high levels of axial or general root loss the difference disappeared almost completely. The presence of RCA and low nodal root numbers both improved nitrogen uptake, irrespective of root loss.

Phosphorus uptake is reduced by all types of root loss

In barley and bean phosphorus uptake decreased sharply under all types of root loss, though the effect of lateral root loss was slightly less pronounced than the effect of axial or general root loss (Figures A.4, A.5). In maize, this pattern was present in all soils except the high nitrogen, low phosphorus soil, although the decrease in phosphorus uptake under lateral root loss was slightly smaller (Figure A.6). In the high nitrogen, low phosphorus soil, the high branching phenotypes did not see their nitrogen uptake amount altered by much under lateral root loss. In all three species greater lateral root branching densities were associated with greater phosphorus uptake, similar to the results from earlier studies [92, 154, 221] and the differences between the uptake amounts of different phenotypes were very large, a difference that was robust under different root loss intensities. In barley, the presence of RCS reduced phosphorus uptake, as found in [175], and phenotypes with greater tiller numbers had greater uptake, both of these findings were true independent of root loss type and intensity. In bean, the presence of RCA had almost no effect on phosphorus uptake and high nodal root numbers were associated with greater phosphorus uptake, again these relationships were independent of root loss. For maize grown in the high-nitrogen, low-phosphorus soil subjected to lateral root loss, phosphorus uptake increased slightly for some of the high branching phenotypes, while it declined for medium or low branching phenotypes.

Nutrient uptake efficiency increases under root loss

We define nutrient uptake efficiency as µmol nutrient taken up from the soil per cm of root surface per day. Nitrogen uptake efficiency increased for almost all phenotypes of all three species in every soil under every type of root loss compared with no root loss (Figures A.7, A.8, A.9). In barley, nitrogen uptake efficiency increased much more under axial or general root loss than under lateral root loss, while in bean and maize the increases were similar for each type of root loss. The effect of root loss on phosphorus uptake efficiency depended on species, phenotype and soil phosphorus content (Figures A.10, A.11, A.12). In barley and maize, phosphorus uptake efficiency increased under axial and general root

loss, while the picture was more mixed in bean, with many phenotypes showing no improvement. In barley, the phosphorus uptake efficiencies of many phenotypes were not affected by lateral root loss, only the low and medium branching phenotypes without RCS showed a decline in all soils. The high branching barley phenotypes with RCS saw their uptake efficiency increase under lateral root loss, although other than that there was little change. In bean, lateral root loss reduced phosphorus uptake efficiency for all phenotypes and soil conditions. In maize, the high branching phenotypes increased uptake efficiency under lateral root loss while the other phenotypes showed small decreases or no effect.

Root length decreases under root loss, even at low intensities

Root length decreased under all types of root loss for all three species, with axial and general root loss associated with larger declines (Figures A.13, A.14, A.15). For barley and bean, root length increased with lateral branching density and axial root number in all soils, irrespective of root loss, while this was not the case for maize. While the presence of RCS increased root length slightly in barley and RCA increased root length in maize, this was not true for bean. In barley and bean the effect of root loss on root length was very similar for all phenotypes, while for maize there were some that showed slightly different trends. For plants subjected to lateral root loss, the amount of root length lost increased with increasing root loss intensity (Figures A.16, A.17, A.18). When plants were subjected to axial or general root loss this relation was less clear and the amount of root length lost in many cases peaked at either the low or medium intensity. The phenotypes that had the greatest root length generally also had the greatest amount of lost root length.

3.5 Discussion

The effect of root loss on simulated plant development depends not only on the type and intensity of root loss, but also on the species, root phenotype and soil nutrient availability. The results from our simulations suggested that root loss is especially detrimental to the uptake of immobile nutrients like phosphorus while lateral root loss has little effect on nitrogen uptake.

Timing and localisation of root loss is more important for plant development than the amount of roots lost

Our results show that the amount of root length lost is not a good predictor of plant performance, relative to the performance without root loss (Figures A.19, A.20, A.21). This makes sense because plants with larger root systems have more roots to lose and therefore will be less affected by the loss of a single root. In a typical root system, there are many roots from which laterals emerge so if laterals are lost they are generally replaced rapidly, which means that plants with large root systems can lose a relatively large amount of root length every day without corresponding reductions in plant function. Many of the simulated phenotypes did not suffer reduced shoot growth even after losing up to 5% of their lateral roots every day, provided phosphorus is not limiting, and some even see an increase in shoot biomass after the loss of lateral roots. Therefore the amount of root length lost as a fraction of the total root length (henceforth referred to as root loss fraction) may be a better predictor of the impact of root loss on shoot dry weight than the absolute amount of roots lost. While there was some correlation between the root loss fraction at the end of the simulation and the reduction in shoot dry weight for bean, this relationship was much weaker for barley and maize (Figures A.22, A.23, A.24). One reason for this is that if a plant loses a large fraction of its root system early during development this can have a big impact on development (Figure 3.10) but the amount of root length lost can be a small fraction of the total root length produced during the whole simulation.

We observed a strong correlation between maximum root loss fraction (the maximum fraction of root system length lost during any of the simulated days) and reduction in shoot dry weight, as compared to the scenario without root loss (Figures 3.7, 3.8, 3.9). This relationship was true for all combinations of soil nutrient availability and root loss types, except for lateral root loss in the high-phosphorus soils, where it was absent in barley and maize and less clear in bean. A high maximum root loss fraction not only means that a plant lost a large fraction of the resources invested in roots at one point during development, but also that the development of its root system was set back several days, if not more,

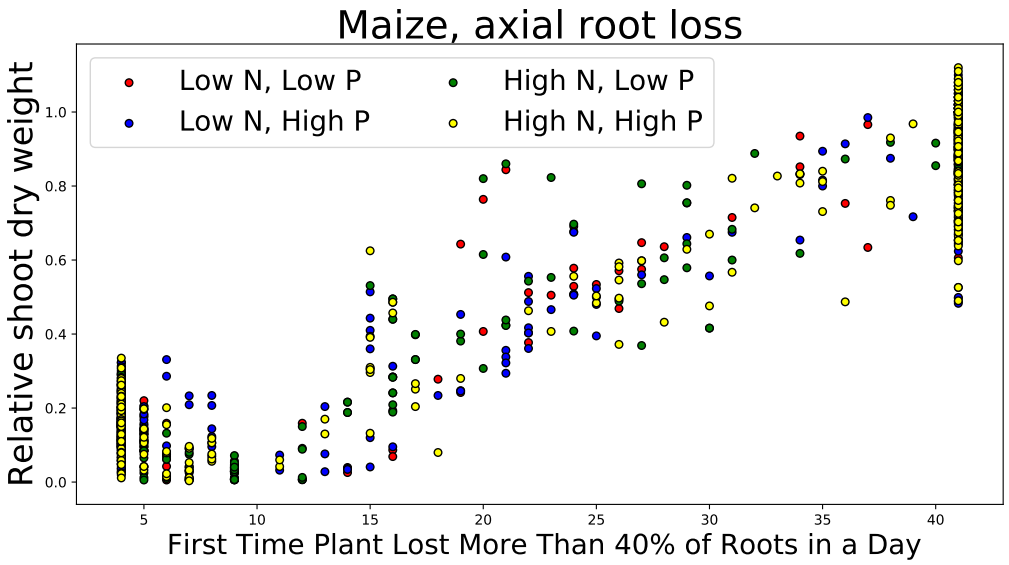


Figure 3.10: Shoot dry weight ratios of plants affected by axial root loss with respect to their shoot dry weight when root loss is absent plotted versus the first day at which these plants lost more than 40% of their root systems in a 24-hour period. The colors indicate soil type. When a plant does not lose more than 40% of their root system during a 24-hour period during the simulation, they are assigned the value 41.

lowering nutrient uptake capacity. This leads to a reduction in shoot growth, leading to lower photosynthesis rates which means less resources are available to grow more roots. Our results show that the impact of such a setback on development is so severe that shoot biomass can be reduced by more than 80%. This explains why axial root loss is so much more detrimental to plant development than lateral root loss; under axial root loss, it is much more likely for a plant to lose a large fraction of its root system, since a single axial root and its lateral roots can represent a large fraction of the root system length. With even a 5% daily loss of lateral roots, this represents a much more constant rate of root loss that is unlikely to lead to large root loss fractions.

The number of axial roots formed early in development determine resilience to axial root loss

Our results suggest that bean is much more resilient to axial root loss than both barley and maize (Figures 3.4, 3.5, 3.6, 3.7, 3.8, 3.9). As we discussed in the previous section, the maximum root loss fraction is a good predictor of the reduction in shoot dry weight due to root loss. Under axial root loss, the maximum root

loss fraction, averaged over all simulations, was 0.76 for barley, 0.46 for bean and 0.63 for maize. Under general root loss these values were 0.76 for barley, 0.53 for bean and 0.67 for maize. This offers an explanation why bean is less susceptible to axial (and general) root loss in our simulations; under randomised root loss, the larger number of axial roots means it is less likely for bean plants to lose a large proportion of their root system in a short period of time.

One might expect large root loss fractions to occur early during development rather than later, since the root system is smaller at that stage and a small number of axial roots and their laterals can represent a large fraction of the root system. While there is some (negative) correlation between maximum root loss fraction and the time at which this maximum root loss fraction occurs for barley and maize, the opposite seems to be true for bean (Figures A.25, A.26, A.27). None of the phenotype-environment combinations we considered for bean had a mean maximum root loss fraction greater than 50% occur before day 15, so we conclude that it is unlikely for bean to lose a large fraction of its root system early, while this is not true for barley and maize. This is because of differences in the number of axial roots that these plants have during the early stages of development: On average, barley had grown 5.0 axial roots at day 10, bean 21.3 and maize 8.7. Differences in early axial root number correspond to differences in how barley, bean and maize root systems develop. First, unlike barley and maize, our bean model includes hypocotyl born roots that emerge regularly during the early stages of development. Second, the basal root whorls in bean have all emerged by day 10 while neither nodal roots nor tiller roots have emerged by this point in barley and only the first whorl of nodal roots has emerged in maize. This means bean has a much larger number of axial roots early in development, making it much more resilient to losing a few axial roots. Finally, in OpenSimRoot, seminal roots emerge as lateral roots from the primary root which means that loss of the primary root implies loss of the entire seminal root system. For the barley and maize phenotypes we considered, the primary root together with the seminal roots are the main axial roots during the first 10 days of development, which makes them vulnerable to root loss.

Differences in axial root numbers between different phenotypes led to differences

in resilience to axial root loss if the differences in axial root numbers occurred early in development. For barley, the phenotypes with 4, rather than 2 or 3 tillers, had extra axial roots, but since the tillers emerged sequentially, this difference in axial roots occurred later in development. The decrease in shoot dry weight due to axial root loss for phenotypes with 2, 3 or 4 tillers, averaged over the other phenes, all environments and all root loss intensities was 56%, irrespective of the number of tillers. For bean and maize, there were differences in axial root numbers earlier in development between different phenotypes. For maize, the different axial root numbers corresponded to different numbers in each root whorl, the first of which emerged 9 days after germination. The maize phenotypes with high axial root number saw a 51% decrease in shoot dry weight, on average, due to axial root loss, versus 55% for the maize phenotypes with low axial root number. For bean, the differences in axial root numbers early in development between phenotypes were the biggest. The bean phenotypes with high axial root number saw a 22% decrease in shoot dry weight, on average, due to axial root loss, versus 29% for the bean phenotypes with low axial root number.

Lateral root loss has little effect on nitrogen uptake but axial root loss does

Since nitrogen in the form of nitrate (the predominate form of available nitrogen in most agricultural soils) is highly mobile, competition among roots for nitrogen capture occurs even at low branching densities. Because of this, we expect the amount of nitrogen gathered from the soil per amount of resources invested to be greater at reduced branching densities, as was confirmed in previous in silico [154, 164] and field [219] studies. This was borne out very clearly for barley and maize, where the densely branching phenotypes performed the worst in the low nitrogen, high phosphorus soil and low numbers of axial roots were associated with greater shoot biomass. For bean, the opposite was true and densely branching phenotypes with medium or high basal root numbers performed the best. This is likely due to the fact that bean roots can fix nitrogen so even if they are competing very strongly for mineral nitrogen in the soil, they are still acquiring nitrogen through fixation. The total nitrogen uptake over the course of the simulation correlates well with the simulated shoot biomass in the low nitrogen, high

phosphorus soil for bean and maize but not for barley. The difference in nitrogen uptake between different phenotypes was very small for barley, perhaps because the longer simulated period of time (80 days instead of 40) means that all the phenotypes deplete most of the soil nitrogen. The nitrogen uptake efficiencies also show that parsimonious phenotypes are a better strategy to obtain nitrogen; phenotypes with low lateral branching densities and low axial root numbers had the greatest uptake efficiencies, in all three species. This is congruent with the proposal that parsimonius root phenotypes are better for N capture [112, 115], and by extension for water capture, since water is also a mobile resource [114].

Nitrogen uptake efficiencies increased under root loss, which is in line with the finding that parsimonious root systems are better adapted for nitrogen uptake. Of course the total nitrogen uptake is also an important factor in plant development and when main axes were lost, nitrogen uptake declined significantly, to the point where plants were not able to develop any significant shoot biomass. So while plants that lose a lot of roots might be able to take up more nitrogen per day per root surface area, they are not able to explore enough soil volume to gather all the nutrients they need for development. Root loss was also associated with more shallow root systems, which are not able to access nitrogen that has leached into deeper soil domains. Since lateral root loss did not affect the simulated shoot biomass of nitrogen-stressed plants, the costs and benefits of this balanced out. The resources invested in these laterals, nutrients and carbon, were lost but the carbon spent on root system maintenance decreased and the nitrogen uptake efficiency increased. This was true for all phenotypes we considered.

In barley, the simulated shoot biomass correlated more strongly with the nitrogen uptake efficiency than with the total nitrogen uptake, indicating that carbon limitations are more important for these plants than nitrogen stress (Figures 3.4, A.1, A.7). For bean, the opposite is true and shoot biomass correlates more with total nitrogen uptake than nitrogen uptake efficiency, which means nitrogen limitations take precedence (Figures 3.5, A.2, A.8). For maize, all three measures correlate, which means the optimal phenotypes are optimising on multiple measures (Figures 3.6, A.3, A.9). These differences might be due to differences in root architecture or due to differences in shoot-root ratio.

Phosphorus stress is compounded by root loss

Due to the immobile nature of phosphorus in the soil, phosphorus uptake is linked strongly to soil exploration, especially in the topsoil where phosphorus availability is greatest [116]. For the same reason, competition between roots is small, which explains why the phenotypes with high lateral branching densities are the best performers in the low phosphorus, high nitrogen soil, again confirming previous in silico [155, 154] and field [221, 92] studies. In barley, the phenotypes with the greatest phosphorus uptake were not the best performers in this soil however; the high branching phenotypes without RCS took up more phosphorus from the soil than those with RCS but had 30% lower shoot biomass. This indicates that the trade-off between less uptake and nutrient remobilisation, as well as reduced root respiration costs, is beneficial in low phosphorus environments, in other words, RCS makes the root system more efficient [175]. In maize and bean, the simulated shoot biomass and total phosphorus uptake in the high nitrogen, low phosphorus soil correlate strongly, while the uptake efficiency is less important. This implies that phosphorus supply is the main factor determining growth, although in the case of bean the phenotypes with the greatest uptake efficiency also acquired the most phosphorus, which explains why the differences in shoot biomass are larger in bean than in the other species.

All types of root loss were associated with significant decreases in phosphorus uptake, in all soils. This contradicts the hypothesis put forward in [184], that root turnover (root loss followed by regrowth) allows for the exploration of greater soil volumes. The trade-off is that while root turnover means losing uptake capacity in certain parts of the soil, it also allows the plant to explore new parts of the soil. However, the model utilised to support this hypothesis [184] did not take into account the carbon costs associated with growing and maintaining roots. In contrast, our results do not show any benefit to root loss in low phosphorus soils, suggesting that the trade-off between giving up existing root length and exploring new soil is not favourable. The carbon costs of maintenance could be low enough relative to the cost of growing new roots so that only a very small amount of new root length can be grown when older roots are lost. Also, the

nutrients invested in a root segment are lost upon root death so if a root does not take up more than the nutrients invested in it before being lost, it is a net loss to the plant. Finally, if the buffer power of a soil is high enough, the soil might not be meaningfully depleted of phosphorus during the simulated timeframe and the benefits of exploring new soil are thus too small.

Optimal root phenotypes are similar with and without root loss

An important question is what our results imply for crop management and breeding. In low-phosphorus soils, root loss should be prevented if possible, considering the significant decline in predicted shoot biomass we observed. Axial root loss is detrimental, regardless of the environment, especially for barley and maize. The effect of lateral root loss depends on species as well as environment. For barley plants grown without phosphorus limitations, lateral root loss does not reduce predicted shoot biomass, so preventing lateral root loss is not beneficial. This is also true for bean and maize in environments where nitrogen is limiting but phosphorus is not. In environments where nutrients are not limiting, lateral root loss should always be prevented in bean, the same is true for most maize phenotypes.

Given the large number of environmental factors influencing root loss, it is rarely possible to prevent it entirely. A relevant question then is which root phenotypes should be selected in order to maximize shoot biomass. Phenotypes with higher lateral branching densities are more resilient to lateral root loss, however this does not mean they are also the phenotypes with the highest shoot biomass in all environments (Figures 3.4, 3.5, 3.6). In most environments the top performers are very similar across the different root loss intensities. Even when different phenotypes respond differently to an increase in lateral root loss intensity, as is true for the high-branching maize phenotypes, this is generally not enough to make them a top performer, since the difference in shoot biomass without any root loss was very large to begin with. Resilience to axial root loss was linked to the number of axial roots during the first 10 days after germination. This was both apparent in differences between species and in differences between phenotypes. The bean phenotypes we simulated have far higher axial root numbers at day 10

and show significantly higher resilience to axial root loss. We conclude that in environments where axial root loss is prevalent, selecting for phenotypes with higher axial root number early during development improves performance. The utility of selecting for higher lateral branching densities in environments with lateral root loss depends on the availability of nitrogen and phosphorus, the difference in resilience is generally smaller than the difference in performance without any root loss. Our results, like previous studies indicate that selection for appropriate root architectural phenotypes can substantially improve crop development [115].

3.6 Further research

While the use of OpenSimRoot enabled us to explore a large range of scenarios, its limitations should be acknowledged. Root growth, respiration, exudation and other parameters have only been measured for the first 80 days for barley and for approximately 40 days for bean and maize. Additionally, OpenSimRoot is not yet capable of simulating the processes relevant for flowering and seed setting, meaning we could not directly predict yields and had to rely on shoot biomass as a proxy for fitness. Longer simulations would be interesting to verify if our findings hold up when simulations are run from germination through to harvest. It would be interesting to study the interactions between root loss and a wider selection of phenes and environments. However, with just 3 types of root loss at 3 intensities, 4 different soils and 3 phenes that were varied, this study represents results from 10800 simulations, each requiring between 2 and 48 hours to complete and each using up to 20 GB of computer memory. Considerable computing resources are needed for more extensive simulation studies.

In order to focus on the influence of root loss on plant nutrient uptake and growth, we assumed that root loss was equally likely for every root of the affected root classes. Modelling more realistic scenarios, such as models where a disease spreads through the root system or predation affects localised regions of the soil, would be an interesting extension of these results. Scenarios where root loss depends on local soil conditions are also possible fruitful future avenues of research. There was also no assumption of physiological mitigation strategies by the plant. Older, thicker and more suberised roots are less vulnerable to a range of stresses that

might lead to root loss. Modelling different root loss rates based on root diameter would introduce an interesting trade-off, where thinner roots are more efficient at taking up nutrients, due to their increased surface to volume ratio, but also more susceptible to root loss. We also did not explore the effects of root loss on plant stability and anchorage.

3.7 Conclusions

We used the functional structural plant model OpenSimRoot to simulate the effects of root loss on the development of barley, common bean and maize. Even though we used a simple model for root loss, assuming randomized root loss with equal probabilities for all roots, and looked at a relatively small number of environments, phenotypes, species, root loss types and root loss intensities, our simulations revealed a complex picture that invites further studies. Our simulations showed that the effect of root loss on plant development depends on environment, root loss type and the root system architecture. Root loss was much more detrimental for plants that were subjected to phosphorus stress than plants that were subjected to nitrogen stress, which makes sense because competition between roots is much greater for mobile resources like nitrogen. Axial root loss was much more detrimental than lateral root loss, which had little or even a positive effect on plants that were not phosphorus stressed. There was a clear relationship between the largest cumulative root loss, as fraction of total root production, a plant had to endure and the associated reduction in shoot dry weight. Common bean was more resilient to axial root loss than barley and maize, which was likely because the common bean root system has more axial roots during the early phases of development. We hope that, outlining the far-reaching consequences root loss can have on plant development, we will motivate additional studies on root loss in the field, which are currently scarce. Apart from the normal difficulties associated with studying roots in the field, studying root loss brings additional challenges because roots need to be tracked over time, and it is challenging to impose specific root loss scenarios, which highlights the importance of modelling studies. By shedding light on this difficult to study process we show how modelling can not only verify our understanding of various processes but also how it points the way to new avenues of research.

Chapter 4

Photosynthesis and Drought

The work described in this chapter was done in collaboration with Ishan Ajmera (Penn State University) who dug into the literature to help decide which models would be appropriate and provided most of the relevant parameters. He also helped test the implemented models and our discussions on the assumptions and early results proved invaluable. We are grateful for his contributions.

One of the most important constraints to growth that plants face is the availability of water. Not only does water make up a large part of a plant's total mass, significant amounts of water are lost through the transpiration associated with photosynthesis. If water is in short supply, plants will respond by closing their stomata, reducing water loss through the leaves as well as photosynthesis rates because this also reduces the supply of carbon dioxide. This is a crucial component of plant behaviour that was not yet implemented in OpenSimRoot. In order to better simulate the effects of drought on plant development, we added a number of models to OpenSimRoot, which we describe in this document. Note that a list of all symbols and constants with references to relevant equations or literature can be found in Appendix B.1.

Currently, the photosynthesis model of OpenSimRoot depends solely on the amount of incoming radiation and the leaf area, as a result the development of simulated plants is not affected by drought. To properly model a drought response, we first need to quantify water stress. Then we need to integrate a stomatal conductance model that makes the simulated plant responsive to water stress. The most important effects of drought-induced stomatal conductance are

a reduction in transpiration and photosynthesis rates, so we need a photosynthesis model that takes into account available CO_2 as well as available light. In order to keep individual models simple, we model mesophyll (and for C4 photosynthesis also bundle sheath) CO_2 and O_2 concentrations. Since the reactions involved in photosynthesis are temperature dependent, we also implement a leaf temperature model. Finally, we implement a diurnal radiation model so that carbon and light limitations happen at different times of the day and there are appropriate leaf temperature variations during the day.

Section 2.1 gives an overview of important concepts relating to photosynthesis, explaining some crucial differences between the C3 and C4 photosynthesis pathways. Section 4.1 details the relevant equations in the C3 photosynthesis model, which includes a model for stomatal conductance and models for leaf gas concentrations. In Section 4.2 we explain some of the constraints that OpenSimRoot put on implementing the models relating to the API and timestepping used. In Section 4.3 we explain the numerical root finder we implemented to deal with the constraints in the previous section. In Section 4.4 we solve the C3 model equations (making a few assumptions) analytically and show it matches the solution found by the numerical root finder. In Section 4.5 we write out the equations for C4 photosynthesis. Section 4.6 contains the model for leaf temperature as well as the temperature dependence of parameters in the C3 and C4 photosynthesis models. Section 4.7 describes a model for calculating the solar radiation based on latitude, date and time. Section 4.8 lists some final implementation issues that had to be resolved. Section 4.10 describes the results of plant simulations under drought using the new models described in this chapter. An overview of all the symbols, constants, variables and a list of all important equations can be found in Tables B.1, B.3 and B.4.

4.1 Modelling C3 photosynthesis

In order to model plant responses to drought we need to have an idea of how plants sense drought. Since the plant hormone abscisic acid (ABA) is produced in drying plant tissues [109] and ABA causes stomata to close [97], it was thought that ABA produced in roots and transported to the leaves was the main path-

way regulating stomatal drought responses. This hypothesis was supported by several experiments [73, 95] but more recent experimental work calls into question whether root-produced ABA is the primary driver of stomatal closure under drought. If leaf turgor is maintained, stomatal closure does not happen even if roots are exposed to low water potentials [37]. Isolating roots and leaves and subjecting both to water stress causes ABA concentrations to increase sharply in leaves but not in roots in Arabidopsis [86]. A large fraction of the ABA found in roots may be produced in leaves [31, 86] or require precursors produced in leaves [125, 167]. There is evidence that ABA produced in leaves is important for the regulation of stomatal conductance. Arabidopsis leaf ABA levels increased and decreased when huminidy was decreased or increased [86, 144] and ABA production increased in leaves that were forcibly dehytrated using pressure chambers [126, 191]. This suggests that leaf ABA concentration increases whenever leaf water content, and hence leaf water potential, decreases [173].

Note that we still do not have a complete mechanistic understanding of stomatal responses to factors that influence plant water status even though evidence suggests stomata respond to leaf water potential [25]. The xylem water flow model described in [45] assumes steady-state flow to calculate the quantity and distribution of water uptake by the roots for a given hydraulic potential at the collar (which is where the hypocotyl and shoot meet). In OpenSimRoot, the collar hydraulic potential is set to the value for which the root water uptake matches the transpiration rates provided by the shoot model (provided the collar potential is between 0 and -15000 hPa). Until the geometry of the shoot system is simulated fully in OpenSimRoot, the collar water potential is the best proxy for leaf water potential available. So we quantify drought stress by mapping the collar water potential to a stress factor, a number between 0 (maximum stress) and 1 (no stress), in line with how nutrient stresses are simulated in OpenSimRoot. We do this by specifying a drought response curve, $S_w(\Psi_c)$, where S_w is the water stress factor and Ψ_c is the collar water potential. This drought response curve is not hard coded but specified in the input file. This allows users to specify different drought responses and sensitivities. The drought response curve we will use is

piecewise linear and defined as follows:

$$S_w(\Psi_c) = \begin{cases} 1 & \text{if } \Psi_c > -4000\text{hPa}, \\ \frac{\Psi_c+14000}{10000} & \text{if } -4000 > \Psi_c > -14000\text{hPa}, \\ 0 & \text{if } \Psi_c < -14000\text{hPa}. \end{cases} \tag{4.1.1}$$

A field potential of -15000 hPa is commonly taken as the wilting point [141], beyond which plants are not able to recover. So we take this as the lowest possible collar hydraulic potential. The numbers above were chosen so that we reach a maximum stress response a bit earlier than the wilting point and the stress response is somewhat gradual. Note that these numbers are not based on measurements or literature values. We will use this stress factor in the model for stomatal conductance, see below. It is also used to reduce shoot growth rates through a stress response curve defined in input files, just like for nutrient stresses. Currently, OpenSimRoot calculates photosynthesis rates through the LINTUL model [182, 158]. The energy cointained in the sunlight intercepted by the leaves, L is calculated as

$$L = a \cdot I_s \cdot f_{PAR}(1 - e^{-k \cdot LAI}), \tag{4.1.2}$$

where a is the area in m^2 each plant has available, I_s is the solar irradiance in W m^2, f_{PAR} the fraction of radiation that is photosynthetically active, k the so-called extinction coefficient and LAI the leaf area index, which is the leaf area divided by the area available to each plant [158, 195]. The final term calculates how much of the potentially available light is actually intercepted by the leaves, taking into account self-shading effects from the canopy. The carbon assimilation rate, P in $\frac{g}{s}$, is then given by

$$P = L\epsilon_{LU}, \tag{4.1.3}$$

where ϵ_{LU} in $\frac{g}{J}$ is the light use efficiency of the plant in question. This simple model is good enough for many purposes but does not include any dependence on plant water status. Seeing as under drought conditions, plants close their stomata, which reduces gas exchange between leaves and the atmosphere, we require a photosynthesis model that takes leaf CO_2 concentrations into account, as well

as light interception. We will implement the Farquhar-Von Caemmerer-Berry model for C_3 photosynthesis [58, 59, 200], as described in detail in [201]. This steady-state model is so influential that it is known as "the canonical model" and most published steady-state photosynthesis models are based on or derived from it [136]. This model combines biochemical and irradiation dependent constraints and has been applied in a wide range of contexts [202]. The photosynthetic assimilation rate A in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ is equal to

$$A = \min\{A_c, A_j, A_p\}, \tag{4.1.4}$$

where A_c in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ is the rubisco-limited rate, A_j in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ the electron transport-limited rate and A_p in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ the phosphate-limited rate of carbon assimilation. We will assume that phosphate is not limiting so ignore A_p from here onward. A_c and A_j are defined as

$$A_c = \frac{V_{cmax}(C_m - \Gamma^*)}{C_m + K_C(1 + \frac{O_m}{K_O})}, \tag{4.1.5}$$

$$A_j = \frac{(C_m - \Gamma^*)J}{4 \cdot C_m + 8\Gamma^*}. \tag{4.1.6}$$

Here V_{cmax} in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ is the maximum rubisco carboxylation rate, K_C and K_O in μmol/mol are the rubisco Michaelis constants for CO_2 and O_2 and Γ^* in $\frac{\mu\text{mol}}{\text{mol}}$ is the CO_2 compensation point without dark respiration (the leaf respiration that happens independently of photosynthesis). C_m and O_m in $\frac{\mu\text{mol}}{\text{mol}}$ are the internal CO_2 and O_2 concentrations, which depend on the stomatal conductance to water, g_w in $\frac{\text{mol}}{\text{m}^2\,\text{s}}$. J in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ is the potential electron transport rate, which depends on the amount of radiation intercepted by the leaves and is defined below. If $C_m \leq \Gamma^*$ then $A_c = A_j = 0$.

We model the internal leaf CO_2 and O_2 concentrations, C_m and O_m, using the following differential equations, containing one term for the diffusion of CO_2/O_2 into/out of leaves through stomata, and one term for the depletion/production of CO_2/O_2 because of photosynthesis and one term for the production/depletion of CO_2/O_2 due to respiration. Assuming that diffusion of these gases is governed by Fick's law [60], for the diffusion term we get the fluxes

$$J_C = \frac{g_w}{1.6}\left(C_A - C_m\right), \tag{4.1.7}$$

$$J_O = \frac{g_w}{1.25}\left(O_A - O_m\right). \tag{4.1.8}$$

The ratios of the diffusivities in air of water, carbon dioxide and oxygen mean that $\frac{g_w}{1.6}$ is the stomatal conductance to carbon dioxide and $\frac{g_w}{1.25}$ is the stomatal conductance to oxygen [76]. The unit of the flux J_C is $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$. In order to write down a differential equation of the form $\frac{\partial C_m}{\partial t} = f(C_m)$, where the left side has units $\frac{\mu\text{mol}}{\text{mol}\,\text{s}}$ we need to include a conversion factor with units $\frac{\text{m}^2}{\text{mol}}$. Using the ideal gas law and the fact that the change in concentration due to a diffusion flux depends on the size of the container diffusion is going into or out of, we multiply the fluxes by $\frac{RT_A}{Pd_L}$, which has units $\frac{\text{m}^3\,\text{Pa}\,\text{K}}{\text{K}\,\text{mol}\,\text{Pa}\,\text{m}} = \frac{\text{m}^2}{\text{mol}}$, where R in $\frac{\text{J}}{\text{K}\,\text{mol}}$ is the ideal gas constant, T_A in K the air temperature, P in Pa the air pressure and d_L in m the thickness of the mesophyll cells. As we will later see, this conversion factor will disappear from the equations so for now we will set $d_L = 100$ µm, slightly lower than leaf thickness after 5 days [98].

Including terms for the photosynthetic assimilation rate A and the dark respiration rate R_d, both in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ we get

$$\frac{\partial C_m}{\partial t} = \frac{R \cdot T_A}{P \cdot d_L}\left(\frac{g_w}{1.6}\left(C_A - C_m\right) - A + R_d\right), \tag{4.1.9}$$

$$\frac{\partial O_m}{\partial t} = \frac{R \cdot T_A}{P \cdot d_L}\left(\frac{g_w}{1.25}\left(O_A - O_m\right) + A - R_d\right). \tag{4.1.10}$$

Here A is given by equation 4.1.4. Since the gas concentrations in leaves reach equilibrium conditions in a few seconds at most, as we will show in Section 4.2, and the relevant timescale for OpenSimRoot is measured in days, we will assume the system is at steady-state. At steady-state, $\frac{\partial C_m}{\partial t} = \frac{\partial O_m}{\partial t} = 0$ and

$$C_m = C_A - \frac{1.6}{g_w}(A - R_d), \tag{4.1.11}$$

$$O_m = O_A + \frac{1.25}{g_w}(A - R_d). \tag{4.1.12}$$

For the stomatal conductance to water g_w, we use a slightly adapted version of the Ball-Berry-Leuning model, described in [104], which states that

$$g_w = \frac{m \cdot S_w(\Psi_c)(A - R_d)}{C_A - \Gamma}\left(1 + \frac{VPD}{VPD_r}\right)^{-1} + g_{w0}. \tag{4.1.13}$$

Here m is an empirical constant, $S_w(\Psi_c)$ is the water stress factor defined in 4.1.1,

Γ in $\frac{\mu\text{mol}}{\text{mol}}$ is the CO_2 compensation point with dark respiration, defined below, see equation 4.1.18, VPD in kPa the vapour pressure deficit, VPD_r in kPa a reference vapour pressure deficit and g_{w0} in $\frac{\text{mol}}{\text{m}^2\,\text{s}}$ the residual conductance. The vapour pressure deficit, VPD is given by

$$VPD = VP_s - VP_a, \tag{4.1.14}$$

where VP_s in kPa is the saturated vapour pressure and VP_a in kPa is the actual vapour pressure. This is related to the relative humidity H_r through

$$H_r = \frac{VP_a}{VP_s}. \tag{4.1.15}$$

The potential electron transport rate, J, which appears in equation 4.1.6, is defined as:

$$J = \frac{I_2 + J_{max} - \sqrt{(I_2 + J_{max})^2 - 4\theta I_2 J_{max}}}{2\theta}, \tag{4.1.16}$$

where I_2 in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ is the energy in the light absorbed by photosystem 2, J_{max} in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ is the maximum electron transport rate and θ is an empirical curvature factor. I_2 is given by

$$I_2 = \frac{\beta I_s \alpha (1 - f)}{2}, \tag{4.1.17}$$

where β in $\frac{\mu\text{mol}}{\text{J}}$ is a conversion factor, I_s in $\frac{\text{W}}{\text{m}^2}$ is the solar irradiation, α the leaf absorbtance and f a factor to correct for the spectral quality of the light.

Finally, equation 4.1.13 contains Γ, the CO_2 compensation point with dark respiration included, at which photosynthesis and respiration rates are equal. Γ is given by

$$\Gamma = \frac{\Gamma^* + \frac{K_C R_d}{V_{cmax}} \left(1 + \frac{O_m}{K_O}\right)}{1 - \frac{R_d}{V_{cmax}}}. \tag{4.1.18}$$

In summary, the C3 photosynthesis rate is given by equation 4.1.4, which states that the assimilation rate is the minimum of the carbon-limited (equation 4.1.5) and light-limited (equation 4.1.6) rates. The assimilation rates depend on the mesophyll carbon dioxide and oxygen concentrations (equations 4.1.11 and 4.1.12)

as well as the solar irradiation. The leaf gas concentrations depend on respiration and photosynthesis rates as well as the stomatal conductance (4.1.13). Stomatal conductance depends on atmospheric parameters, photosynthetic assimilation rates and the water status of the plant, as quantified by the water stress factor (equation 4.1.1). This is a nonlinear system of equations, the solution to which will give us the photosynthesis rate, our primary variable of interest.

4.2 OpenSimRoot Implementation

When implementing new models in OpenSimRoot, we should remain within the OpenSimRoot design framework where possible and should try to work with what is present in the application programming interface (API). This means that anything implemented should be backwards compatible, models should be implemented in a modular way such that each individual model can be swapped out for a different one and information should generally travel in one direction (this means information is passed between models by request). As we shall see, this caused some difficulties for implementation which were overcome through the addition of new functionality to the OpenSimRoot engine.

Models, or plugins, in OpenSimRoot fall in two broad categories. The first are variables defined by algebraic relationships. That is, at any given time t, $y(t) = F(x_1(t), x_2(t), ..., x_n(t))$ for some function F and other variables $x_1, ..., x_n$. These variables are added to the simulation by including appropriate SimulaDerivative XML tags in the input file. The second category is of variables that are defined by differential equations, these are calculated by integrating rates over time, using a Runge-Kutta predictor-corrector method. To add variables of this type, the SimulaVariable XML tag is usually used.

In a typical simulation there will be a number of cyclical dependencies between variables. For example, growth rates depend on the photosynthesis rate, which depends on leaf area, which depends on growth rates. If SimulaDerivatives depend on each other cyclically, they will get stuck in a perpetual loop of requesting a value from each other, resulting in a segmentation fault. If there is a SimulaVariable in a cyclical chain of dependencies, the SimulaVariable will be predicted

forward in time, then the other variables in the chain will be calculated with this predicted variable, after which the predicted value is updated.

To determine photosynthesis rates in the new C3 model described in Section 4.1, we need to find the steady state solutions or equations 4.1.9 and 4.1.10. This is equivalent to finding the solution to the system of equations consisting of equations 4.1.4, 4.1.5, 4.1.6, 4.1.11, 4.1.12, 4.1.13 and 4.1.1. Since there are cyclical dependencies between these equations, e.g. the photosynthesis rate and leaf internal CO_2 concentration depend on each other, we have to represent one of the variables with a SimulaVariable to avoid the simulation getting stuck in an endless loop. The internal CO_2 concentration is an obvious candidate to represent as a SimulaVariable. The rate of change would be given by differential equation 4.1.9. However, the timescale implied by this equation does match the timescale relevant to OpenSimRoot. Time in OpenSimRoot is expressed in units of days, with a timestep typically being 0.1 days, while the CO_2 concentration in leaves converges to the steady-state in just a few seconds, as can be seen from Figure 4.1. This figure shows the result of numerically integrating equation 4.1.9 using the forward Euler method.

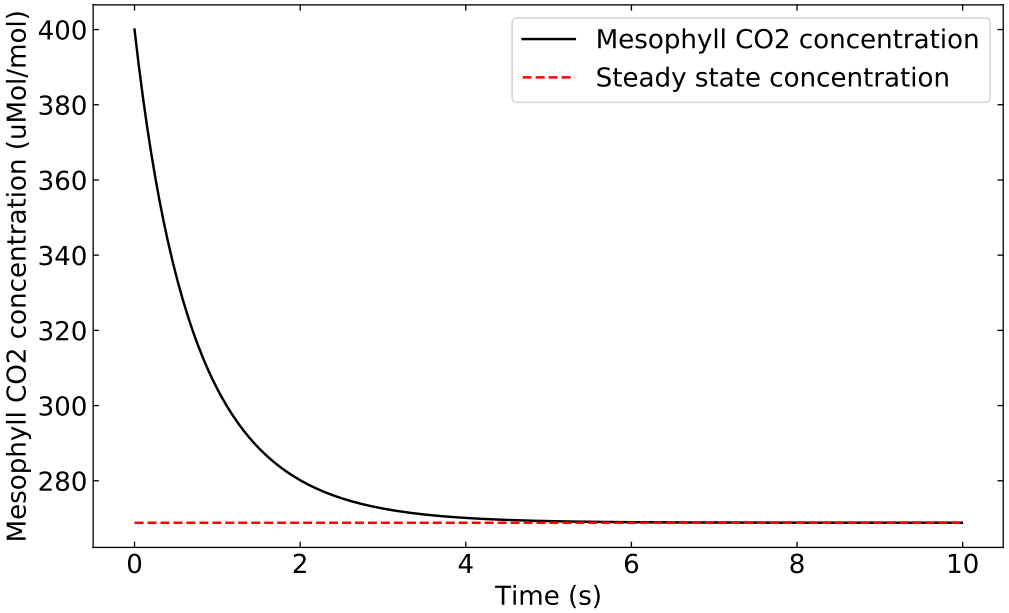


Figure 4.1: Evolution of the mesophyll CO_2 concentration numerically integrated using the forward Euler method.

In an early implementation, this mismatch in timescales caused the gas concentra-

tions to overshoot massively in the first timestep. Then in the following timestep, the value overshot the other way. This continued until they eventually diverged to infinity. Because the system converges to the steady state solution in a fraction of a typical OpenSimRoot timestep, we assume the system is always in steady state.

After writing down the solution to the system of of equations 4.1.4, 4.1.5, 4.1.6, 4.1.11, 4.1.12, 4.1.13 and 4.1.1 analytically we found that implementing this in OpenSimRoot leads to a number of issues. It would require different OpenSimRoot submodels to have the same underlying equations, in contradiction to the modular design of OpenSimRoot and it does not allow for temperature models to be included. Because of this, we updated the OpenSimRoot engine so that the solution to the system of equations 4.1.4, 4.1.5, 4.1.6, 4.1.11, 4.1.12, 4.1.13 and 4.1.1 is calculated numerically.

4.3 Numerical root finder

To explain the numerical root finder we implemented, we will first explain some parts of the OpenSimRoot API in detail. In OpenSimRoot, the methods that calculate the values of state variables are generally not given any input values in their function call besides the (simulated) time at which the variable is requested. These functions in turn request the values of the state variables they depend on at the same (simulated) time in their code. In addition, when the value of a state variable is requested, the method corresponding to that state variable is not called directly. Instead, the **get** method of the parent class (in our case **SimulaDerivative**) is called. This allows OpenSimRoot to cut down on calculations by saving values in a cache and allows for OpenSimRoot to interpolate or know when to integrate forward in time (used by SimulaTables and SimulaVariables). This **get** method is what we will alter to implement our Newton solver. The (simplified) pseudocode for the SimulaDerivative::get method was as follows:

```
1 get(t, x){
2   if time t is close to cached time
3     return cached value
4   x = calculate(t)
5   set cached value to x and cached time to t
6 }
```

Here "calculate(t)" means that the calculate method of the class responsible for the state variable in question is called. For example, for the state variable "root segment dry weight", this will (usually) be the class called "RootSegmentDry-Weight" in "RootSegmentDryWeight.cpp". The "calculate(t)" method requests the volume and density at time t and multiplies them to get the dry weight.

If SimulaDerivatives depend on each other, possibly indirectly, they will get stuck in an infinite loop, perpetually calling each others **calculate** methods (through their **get** methods). We want instead to pick a 'reasonable' starting value for one of them and then to iterate using the Newton-Rhapson method until we have found a solution to the system of equations 4.1.4, 4.1.5, 4.1.6, 4.1.11, 4.1.12, 4.1.13 and 4.1.1 (the steady state). In the one-dimensional case, where we have an equation $x = F(x)$ and we are trying to find a solution x^* such that $x^* = F(x^*)$, we can turn the fixed point equation $x = F(x)$ into the following form $F(x) - x = 0$. In other words, we will use the Newton-Raphson method to find a root of the expression $F(x) - x$. This means that, for a given x_n, x_{n+1}, the next iteration is defined as

$$x_{n+1} = x_n - \frac{F(x_n) - x_n}{F'(x_n) - 1}. \tag{4.3.1}$$

The pseudocode looks like this:

```
get(t, x){
  if time t is close to cached time
    return cached value
  if method was (indirectly) called by itself
    if a callback cache was set
      return cached callback value
    else
      set callback cache to default value
      return default value
  x = calculate(t)
  if calculate led to a callback loop
    while |x - cached callback value| > tolerance
      set callback cache to x
      y1 = calculate(t)
      set callback cache to x + dx for some small value of dx
```

```
16        y2 = calculate(t)
17        yslope = dy/dx = (y2 - y1)/dx
18        x = x - (y1 - x)/(yslope - 1)
19    set cached value to x and cached time to t
20 }
```

Let us walk through what would happen for a SimulaDerivative whose value (indirectly) depends on itself through a function $x = F(x)$. When it's value is requested for a new time t, the condition at line 2 is not satisfied and neither is the condition at line 4 (yet). So it will proceed to line 10, where the relevant `calculate` method will be called. Through the dependencies in the calculate method it will (indirectly) call itself and then will satisfy the condition in line 4. Since no callback cache has been set yet, it will set the callback cache to a default value and return. Then, returning to the first frame, it will execute lines 11 and 12 and satisfy the condition in line 13. Now, while the difference between the value in the callback cache and x_0, the value returned by the `calculate`, method is larger than the tolerance, it will run through the following steps: The callback cache will be set to x_0, which is the value returned by the `calculate` method. Then `calculate` will be called to calculate $F(x_0)$. Then the callback cache will be set to $x_0 + \delta$ for $\delta = 10^{-4}$ and then `calculate` will be called to calculate $F(x_0 + \delta)$. With this we can calculate $x_1 = x_0 - \frac{F(x_0)-x_0}{F'(x_0)-1}$. These steps will be repeated until $x_{n+1} = x_n$, which is the solution we are looking for.

It often happens that there is a situation where we have two variables x and y that are related through the set of equations

$$x = F(x, y), \tag{4.3.2}$$

$$y = G(x, y). \tag{4.3.3}$$

What will the pseudocode described above do? In OpenSimRoot, one of these variables, x, will call back to itself first in an "outer loop". It will set a callback cache value, x_0. Then when "unwinding" the outer loop, x_0 will be returned to the method that calculates y. Since y also depends on itself, it will call back to itself (possibly indirectly), and form an "inner loop". After setting a cached value y_0 and returning to the first frame where the calculation of y was called,

the Newton-Raphson method will be used to find the value y_0^* that satisfies

$$y_0^* = G(x_0, y_0^*). \tag{4.3.4}$$

This y_0^* will then be returned to the code that calculates x. The Newton-Raphson method will then be started for x, but for every step, the code will run through an entire loop to find an y_n^* that satisfies

$$y_n^* = G(x_n, y_n^*). \tag{4.3.5}$$

Here x_{n+1} is given by

$$x_{n+1} = x_n - \frac{F(x_n, y_n^*) - x_n}{\partial_x F'(x_n, y_n^*) - 1}. \tag{4.3.6}$$

Note that if we use the pseudocode as written above, the intermediate values $y_0^*, ..., y_n^*$ would be saved in the cache, which we do not want to do until x has converged to its fixed-point value. So we add a static variable (a static variable is shared among all objects belonging to the same class and all derived classes) to keep track of which variable is the first to call back to itself and a static variable that we use to detect if we are in a callback loop or not. This allows us to only write to the cache once we have completely unwound all callback loops and found the fixed point of all equations involved. Bounds on the final values returned by individual models prevent unphysical solutions, for example those with negative concentrations, from being selected.

If the system does not converge to a solution in 40 iterations, it is unlikely that any further iterations will lead to convergence. It is possible that $x_n = x_{n-i}$ for some $i > 0$, for example if the sequence keeps alternating between two values without converging to the fixed point. So after 40 iterations, we try multiplying x_n by a number between 0 and 2 every 41 iterations. In some cases, this can move it to the basin of attraction of the fixed point. If after 240 iterations there still is no convergence, we estimate the value as either the value at the previous time or as the default value (if specified). The final pseudocode is

```
get(t, x){
  if time t is close to cached time
    return cached value
```

```
  if method was (indirectly) called by itself
    if the starter of the callback loop has not been set yet
      set this object as the callback loop starter
    if a callback cache was set
      return cached callback value
    else
      set callback cache to default value
      return default value
  x = calculate(t)
  if calculate led to a callback loop
    while |x - cached callback value| > tolerance
      if needing more than 40 iterations
        try perturbation every 41 iterations
      if needing more than 240 iterations
        estimate x
      set callback cache to x
      y1 = calculate(t)
      set callback cache to x + dx for some small value of dx
      y2 = calculate(t)
      yslope = dy/dx = (y2 - y1)/dx
      x = x - (y1 - x)/(yslope - 1)
  if  we are not in a callback loop anymore and this object is
   the callback loop starter
    set cached value to x and cached time to t
}
```

The actual code can be found in the OpenSimRoot repository in the file name SimulaDerivative.hpp, located in `OpenSimRoot/src/engine`. Note that at the time of reading, this new code might not have been merged into the public OpenSimRoot repository yet, because it will only made public once an article utilising this new functionality is ready for publication.

Using this numerical solver, OpenSimRoot calculates the simultaneous solution to equations 4.1.4, 4.1.5, 4.1.6, 4.1.11, 4.1.12, 4.1.13 and 4.1.1, which comprise our new photosynthesis and drought model. It is automatically backwards compatible, which is important to guarantee that old input files produce the same result as with older OpenSimRoot versions, because this new code is only active in cases that would have caused older versions of OpenSimRoot to crash.

In the following section we analytically calculate a close approximation to the solution to the C3 model equations and show that our numerical root finder converges to the same solution for a range of different solar irradiation values.

4.4 Analytically solving C3 model equations

Recall the equation describing mesophyll oxygen concentration:

$$O_m = O_A + \frac{1.25}{g_w}(A + R_d) \tag{4.4.1}$$

The atmospheric oxygen concentration, O_A, is generally around 210 $\frac{\text{mmol}}{\text{mol}}$. Because the maximum rubisco carboxylation rate of maize at 25 °C is 49 $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ [216], the photosynthetic assimilation rate is usually below $A = 50\ \frac{\mu\text{mol}}{\text{m}^2\,\text{s}} = 0.05\ \frac{\text{mmol}}{\text{m}^2\,\text{s}}$ during the day. The dark respiration of maize at 25 °C is approximately $R_d = 1.95\ \frac{\mu\text{mol}}{\text{m}^2\,\text{s}} = 0.00195\ \frac{\text{mmol}}{\text{m}^2\,\text{s}}$ [216] . With stomatal conductances typically around $g_w = 0.1\ \frac{\text{mol}}{\text{m}^2\,\text{s}}$, this means that O_A is at least 50 times larger than $\frac{1.25}{g_w}(A + R_d)$. So we make the approximation

$$O_m \approx O_A. \tag{4.4.2}$$

Now we recall the relevant model equations:

$$A_c = \frac{V_{cmax}(C_m - \Gamma^*)}{C_m + K_C(1 + \frac{O_m}{K_O})} \tag{4.4.3}$$

$$A_j = \frac{(C_m - \Gamma^*)J}{4 \cdot C_m + 8\Gamma^*} \tag{4.4.4}$$

$$A = \min\{A_c, A_j\} \tag{4.4.5}$$

$$C_m = C_A - \frac{1.6}{g_w}(A - R_d) \tag{4.4.6}$$

$$g_w = \frac{m \cdot S_w \cdot (A - R_d)}{C_A - \Gamma}\left(1 + \frac{VPD}{VPD_r}\right)^{-1} + g_{w0} \tag{4.4.7}$$

Assuming for the moment that carbon is limiting, so $A = A_c$, and leaf temperature is equal to the air temperature (so not dependent on stomatal conductance),

we find the steady-state solution by writing:

$$\begin{aligned}0 =& \frac{g_w}{1.6}\left(C_A - C_m\right) - \left(A - R_d\right)\\ =& \left(\frac{m \cdot S_w \left(C_A - C_m\right)}{C_A - \Gamma}\left[1 + \frac{VPD}{VPD_r}\right]^{-1} - 1.6\right)\left(\frac{V_{cmax}(C_m - \Gamma^*)}{C_m + K_C(1 + \frac{O_m}{K_O})} - R_d\right)\\ &+ g_{w_0}(C_A - C_m). \end{aligned} \tag{4.4.8}$$

Now using our approximation that $O_m = O_A$ and substituting

$$f(VPD) = \left[1 + \frac{VPD}{VPD_r}\right]^{-1}, \tag{4.4.9}$$

$$\kappa = K_C\left(1 + \frac{O_m}{K_O}\right) \approx K_C\left(1 + \frac{O_A}{K_O}\right), \tag{4.4.10}$$

we get

$$\begin{aligned}0 =& \left(\frac{m \cdot S_w \left(C_A - C_m\right)}{C_A - \Gamma} f(VPD) - 1.6\right)\left(\frac{V_{cmax}(C_m - \Gamma^*)}{C_m + \kappa} - R_d\right)\\ &+ g_{w_0}(C_A - C_m)\\ =& \; aC_m^2 + bC_m + c, \end{aligned} \tag{4.4.11}$$

with

$$a = \frac{m \cdot S_w(R_d - V_{cmax})}{C_a - \Gamma} f(VPD) - g_{w_0}, \tag{4.4.12}$$

$$\begin{aligned}b =& \; g_{w_0}\left(C_A - \kappa\right) - 1.6(V_{cmax} - R_d)\\ &+ \frac{m \cdot S_w}{C_a - \Gamma} f(VPD)\left(V_{cmax}(C_A + \Gamma^*) + R_d\left[\kappa - C_A\right]\right), \end{aligned} \tag{4.4.13}$$

$$c = g_{w_0} \cdot C_A \cdot \kappa - \left(\frac{m \cdot S_w \cdot C_A}{C_A - \Gamma} f(VPD) - 1.6\right)\left(V_{cmax} \cdot \Gamma^* + R_d \cdot \kappa\right). \tag{4.4.14}$$

This is a quadratic equation so the the two solutions can be written as

$$C_m = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}. \tag{4.4.15}$$

If, on the other hand, we assume that light is limiting, so $A = A_j$, we get

$$\begin{aligned}0 =& \left(\frac{m \cdot S_w \left(C_A - C_m\right)}{C_A - \Gamma} f(VPD) - 1.6\right)\left(\frac{(C_m - \Gamma^*)J}{4C_m + 8\Gamma^*} - R_d\right) + g_{w_0}(C_A - C_m)\\ =& \; aC_m^2 + bC_m + c, \end{aligned} \tag{4.4.16}$$

which is again a quadratic equation with

$$a = \frac{m \cdot S_w}{C_A - \Gamma} f(VPD)\,(4R_d - J) - 4g_{w0}, \tag{4.4.17}$$

$$b = \frac{m \cdot S_w}{C_A - \Gamma} f(VPD)\Big(C_A(J - 4R_d) + \Gamma^*(J + 8R_d)\Big)$$

$$+ 1.6(4R_d - J) + g_{w0}(4C_A - 8\Gamma^*), \tag{4.4.18}$$

$$c = 1.6\Gamma^*(J + 8R_d) - \frac{m \cdot S_w \cdot C_A}{C_A - \Gamma} f(VPD)\Gamma^*\Big(J + 8R_d\Big) + 8g_{w0}C_A\Gamma^*. \tag{4.4.19}$$

We of course want to make sure that our numerical solver described in Section 4.3 finds a solution that is consistent with the approximate analytic solution derived here. Figure 4.2 show that the numerical solution to equations 4.1.4, 4.1.5, 4.1.6, 4.1.11, 4.1.12, 4.1.13 and 4.1.1 as computed by the numerical root finder described in Section 4.3 is equal to the largest of the four analytic solutions described in this section, across a range of different irradiation values. This makes sense because larger photosynthesis rates correspond to smaller mesophyll carbon concentrations.

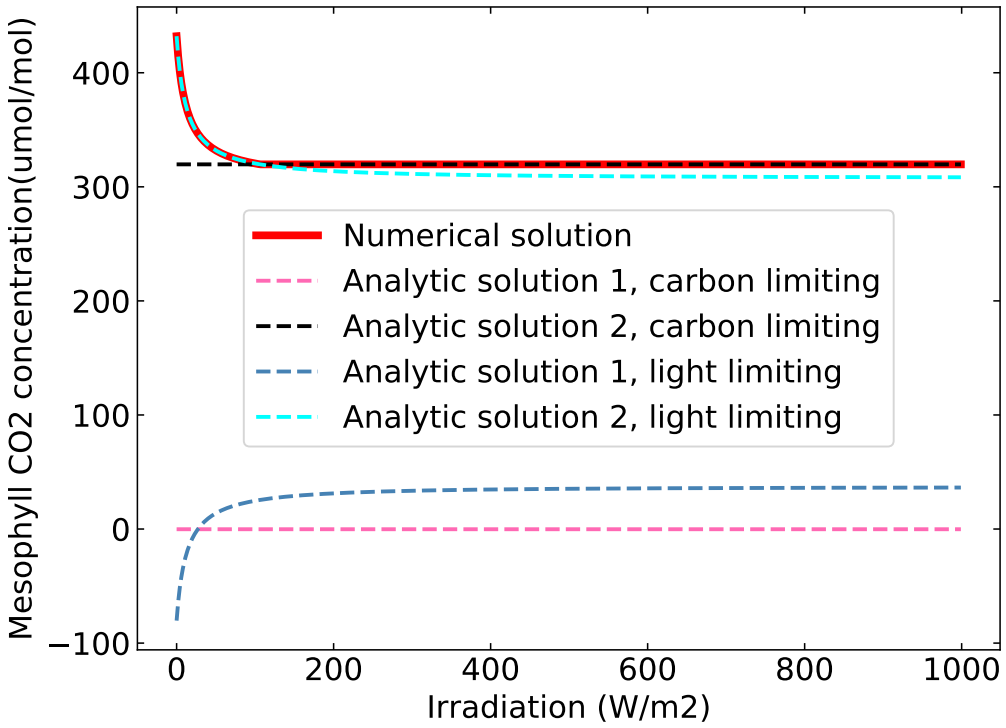


Figure 4.2: Comparing the mesophyll concentrations predicted by the numerical root finder with the four roots derived analytically in this section for a range of different irradiation values.

If we do not make the assumption $O_m = O_A$, but instead have $O_m = O_A + \frac{1.25}{g_w}(A + R_d)$, the two solutions differ slightly, 319.69733 versus 319.69624 $\frac{\mu\text{mol}}{\text{mol}}$.

4.5 Modelling C4 Photosynthesis

In Section 2.1 we explained some of the important differences between C3 and C4 photosynthesis and why C4 photosynthesis has a higher water use and nitrogen efficiency and photorespiration is less of a problem. This makes C4 photosynthesis better when water availability is low and temperatures are high. The extra steps involved in C4 photosynthesis make it less efficient than C3 photosynthesis when water is plentiful and temperatures are low. Approximately 95% of plant biomass on earth comes from plants that fix carbon through C_3 photosynthesis. This includes food crops such as rice, wheat, soybeans and barley. Food crops that use C_4 photosynthesis include species such as maize, sugar cane and sorghum.

As explained in Section 2.1, in C_4 photosynthesis, carbon dioxide is fixed in a two-step process. First, carbon dioxide is fixed by PEP carboxylase in mesophyll cells near the stomata in the form of malate or aspartate. These molecules are then transported to bundle sheath cells where they are decarboxylated. This leads to an environment rich in carbon dioxide, which allows RuBisCO to fix the CO_2 while keeping photorespiration rates low. So for C4 photosynthesis we will add equations for the bundle sheath CO_2 and O_2 concentrations.

We use the following expressions for C4 light-limited and carbon-limited photosynthesis rates [201]:

$$A_c = \frac{(C_s - \frac{1}{2S_{c/o}}O_s)V_{cmax}}{C_s + K_C(1 + \frac{O_s}{K_O})}, \tag{4.5.1}$$

$$A_J = \frac{\left(C_s - \frac{1}{2S_{c/o}}O_s\right)(1-x)J}{3C_s + 7\frac{1}{2S_{c/o}}O_s}, \tag{4.5.2}$$

$$A = \min\{A_c, A_j\}. \tag{4.5.3}$$

These are similar to equations 4.1.5 and 4.1.6, but Γ^*, the CO_2 compensation point without dark respiration (e.g. the CO_2 concentration at which photo-

synthesis and the respiration associated with it are in equilibrium), has been rewritten as $\frac{1}{2S_{c/o}}O_s$, with $S_{c/o}$ the RuBisCo specificity for CO_2 (relative to that for O_2). Also, instead of total electron transport rate J, we use the electron transport rate in bundle sheath cells $(1-x)J$ where x is the fraction of electron transport happening in the mesophyll cells and the constants in the numerators of equations 4.1.6 and 4.5.2 differ slightly because of differences in the energy requirement for whole chain electron transport. We also use the bundle sheath CO_2 and O_2 concentrations, C_s and O_s in $\frac{\mu\text{mol}}{\text{mol}}$, rather than mesophyll concentrations, since assimilation happens in the bundle sheath. Using equations 4.5.1, 4.5.2 and making some simplifying assumptions, Von Caemmerer derived equations for the carbon-limited and light-limited assimilation rates that only require the mesophyll CO_2 and O_2 concentrations [201]. The derivation is similar to the derivation we did for C3 photosynthesis in Section 4.4, but the mesophyll gas concentrations take the place of the atmospheric gas concentrations. Instead of using these equations, we opt for a more modular, easy to understand approach that requires fewer simplifying assumptions. We will simulate both the bundle sheath and mesophyll CO_2 and O_2 concentrations. We start by writing down our differential equations governing their evolution:

$$\frac{\partial C_m}{\partial t} = \frac{R \cdot T_A}{P \cdot d_L}\left(\frac{g_w}{1.6}(C_A - C_m) + g_s(C_s - C_m) - V_P + R_m\right), \tag{4.5.4}$$

$$\frac{\partial C_s}{\partial t} = \frac{R \cdot T_A}{P \cdot d_L}\left(V_P - A_r - g_s(C_s - C_m) + R_s\right), \tag{4.5.5}$$

$$\frac{\partial O_m}{\partial t} = \frac{R \cdot T_A}{P \cdot d_L}\left(\frac{g_w}{1.25}(O_A - O_m) + 0.047 g_s(O_s - O_m) - R_m\right), \tag{4.5.6}$$

$$\frac{\partial O_s}{\partial t} = \frac{R \cdot T_A}{P \cdot d_L}\left(A_r - R_s - 0.047 g_s(O_s - O_m)\right), \tag{4.5.7}$$

where R_m and R_s in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ are the respiration rates in mesophyll and bundle sheath cells, respectively, g_s in $\frac{\text{mol}}{\text{m}^2\,\text{s}}$ is the bundle sheath conductance to CO_2, $0.047 g_s$ is the bundle sheath conductance to O_2 [201], V_P in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ is the PEP carboxylation rate and A_r in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ is the assimilation rate which does not include dark respiration. R_m and R_s are given by

$$R_m = r_m R_d, \tag{4.5.8}$$

$$R_s = (1 - r_m) R_d, \tag{4.5.9}$$

where r_m is the fraction of dark respiration happening in the mesophyll cells. The equations concerning mesophyll concentrations have terms for gas exchange with the atmosphere and bundle sheath cells and a respiration term, while the equations concerning bundle sheath concentrations have a term for the gas exchange with the mesophyll cells, a term for respiration and a term for photosynthesis. PEP carboxylation rates are given by

$$V_p = \min\left\{\frac{C_m V_{pmax}}{C_m + K_P}, V_{pr}\right\}. \tag{4.5.10}$$

Here V_{pmax} in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ is the maximum PEP carboxylation rate, K_P in $\frac{\mu\text{mol}}{\text{mol}}$ the Michaelis constant for PEP carboxylation and V_{pr} in $\frac{\mu\text{mol}}{\text{m}^2\,\text{s}}$ the PEP regeneration rate. Just as in the case of C3 photosynthesis, we note that gas concentrations reach equilibrium values very quickly as compared to the timescale relevant to OpenSimRoot and relative to changes in environmental conditions as can be seen from figure 4.3.

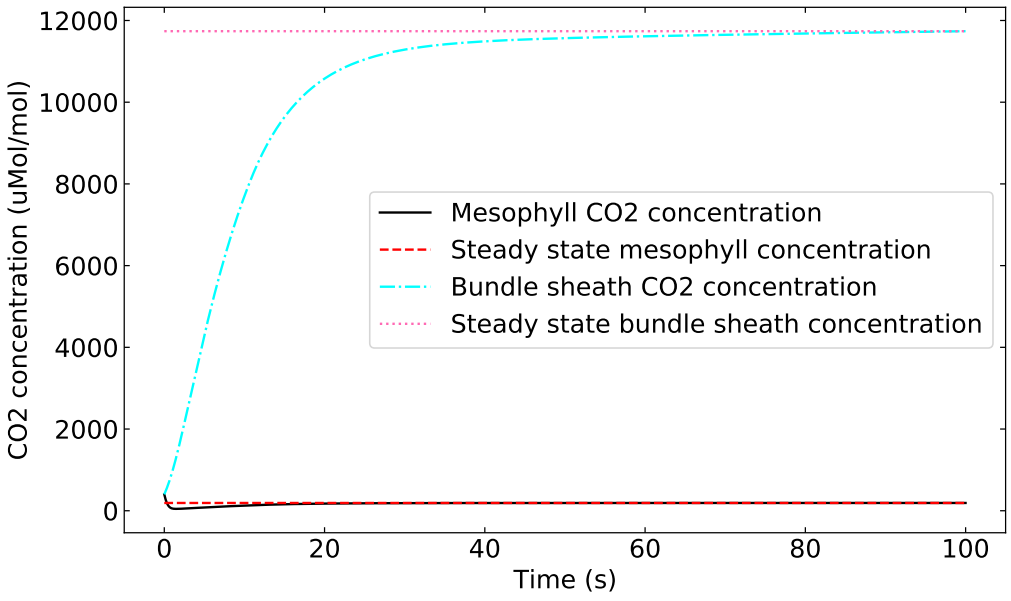


Figure 4.3: Evolution of the mesophyll and bundle sheath CO_2 concentrations numerically integrated using the forward Euler method.

Once again, because the convergence of leaf gas concentrations happens very quickly, relatively to the major processes in OpenSimRoot, we assume the leaf gas concentrations are in steady-state, so we get:

$$C_m = \frac{\frac{g_w}{1.6} C_A + g_s C_s - V_P + R_m}{\frac{g_w}{1.6} + g_s}, \tag{4.5.11}$$

$$C_s = \frac{V_P - A_r + R_s}{g_s} + C_m, \tag{4.5.12}$$

$$O_m = \frac{\frac{g_w}{1.25} O_A + 0.047 g_s O_s - R_m}{\frac{g_w}{1.25} + 0.047 g_s}, \tag{4.5.13}$$

$$O_s = \frac{A_r - R_s}{0.047 g_s} + O_m. \tag{4.5.14}$$

We use the same expression for stomatal conductance as in the case of C3 photosynthesis (equation 4.1.13) but need to use a different expression for the CO_2 compensation point with dark respiration, Γ, since carbon fixation is not happening in the mesophyll cells but in the bundle sheath cells. From [201] we get the C4 expression, which is

$$\Gamma = \frac{K_p}{V_{pmax}} (g_s \Gamma_s - R_m). \tag{4.5.15}$$

Here Γ_s in $\frac{\mu mol}{mol}$ is the bundle sheath CO_2 concentration at the compensation point, which is given by

$$\Gamma_s = \frac{\frac{1}{2S_{c/o}} O_m + \frac{K_C R_d}{V_{cmax}} \left(1 + \frac{O_m}{K_O}\right)}{1 + \frac{R_d}{V_{cmax}}}. \tag{4.5.16}$$

After implementing these equations in OpenSimRoot, the numerical root finder will find the set $\{C_m, C_s, O_m, O_s, V_p, A_c, A_j, A, g_w\}$ that satisfies equations 4.5.11, 4.5.12, 4.5.13, 4.5.14, 4.5.10, 4.5.1, 4.5.2, 4.5.3 and 4.1.13.

4.6 Leaf temperature model

Like many other aspects of plant development and functioning, such as root elongation and respiration rates, photosynthetic assimilation rates are temperature dependent [49, 106]. In order to make accurate predictions in the context of climate change it is important that we can model the effects of elevated temperature and increasing atmospheric CO_2 concentrations. This requires us to model the temperature dependence of the kinetic parameters that our models rely on [22, 107, 128, 204]. For most parameters, we follow [201] and use an Arrhenius function of the form

$$P(T_L) = P_{25} e^{\frac{E(T_L - T_{ref})}{T_{ref} R T_L}}, \tag{4.6.1}$$

where T_L is the leaf temperature in K, T_{ref} the reference temperature, 25 °C or 298.15 K, $P(T_L)$ is the value of the parameter at T_L, P_{25} is the parameter value at 25 degreeCelsius (298.15 K), E in $\frac{\text{J}}{\text{mol}}$ is the activation energy and R is the universal gas constant. We model the temperature dependence of the following parameters with this equation:

$$\Gamma^* = \Gamma^*_{25} e^{\frac{E_{\Gamma^*}(T_L - T_{ref})}{T_{ref} \cdot R \cdot T_L}}, \tag{4.6.2}$$

$$K_C = K_{C25} e^{\frac{E_{K_C}(T_L - T_{ref})}{T_{ref} \cdot R \cdot T_L}}, \tag{4.6.3}$$

$$K_O = K_{O25} e^{\frac{E_{K_O}(T_L - T_{ref})}{T_{ref} \cdot R \cdot T_L}}, \tag{4.6.4}$$

$$R_d = R_{d25} e^{\frac{E_{R_d}(T_L - T_{ref})}{T_{ref} \cdot R \cdot T_L}}, \tag{4.6.5}$$

$$V_{cmax} = V_{c25} \cdot e^{\frac{E_{V_c}(T_L - T_{ref})}{T_{ref} \cdot R \cdot T_L}}, \tag{4.6.6}$$

$$K_P = K_{P25} e^{\frac{E_{K_P}(T_L - T_{ref})}{T_{ref} R T_L}}, \tag{4.6.7}$$

$$\frac{1}{2S_{c/o}} = \frac{1}{2S_{c/o25}} e^{\frac{E_{S_{c/o}}(T_L - T_{ref})}{T_{ref} R T_L}}. \tag{4.6.8}$$

For other parameters, the peaked Arrhenius function of the form

$$P(T_L) = P_{25} e^{\frac{E(T_L - T_{ref})}{T_{ref} R T_L}} \frac{1 + e^{\frac{T_{ref} S - D}{T_{ref} R}}}{1 + e^{\frac{T_L S - D}{R T_L}}} \tag{4.6.9}$$

is used because it better represents the temperature dependence of that parameter. Here D in $\frac{\text{J}}{\text{mol}}$ is the deactivation energy and S in $\frac{\text{J}}{\text{K mol}}$ is called the entropy factor. The following parameters follow this temperature dependence relation:

$$J_{max} = J_{max25} e^{\frac{E_{J_{max}}(T_L - T_{ref})}{T_{ref} R \cdot T_L}} \frac{1 + e^{\frac{T_{ref} S_{J_{max}} - H}{T_{ref} R}}}{1 + e^{\frac{S_{J_{max}} \cdot T_L - D_{J_{max}}}{R \cdot T_L}}}, \tag{4.6.10}$$

$$V_{p25} = V_{pmaxref} e^{\frac{E_{V_{pmax}}(T_L - T_{ref})}{R T_L T_{ref}}} \frac{1 + e^{\frac{S_{V_{pmax}} T_{ref} - D_{V_{pmax}}}{T_{ref} R}}}{1 + e^{\frac{S_{V_{pmax}} T_L - D_{V_{pmax}}}{T_L R}}}, \tag{4.6.11}$$

$$g_s = g_{s25} e^{\frac{E_{g_s}(T_L - T_{ref})}{R T_L T_{ref}}} \frac{1 + e^{\frac{S_{g_s} T_{ref} - D_{g_s}}{T_{ref} R}}}{1 + e^{\frac{S_{g_s} T_L - D_{g_s}}{T_L R}}}. \tag{4.6.12}$$

Figure 4.4 shows an example of the temperature-dependent scaling applied to parameters by these Arrhenius functions. Of course the exact slope and, in the case

of the peaked Arrhenius function, location of the peak, depends on the relevant temperature scaling parameters (activation and deactivation energy and entropy term).

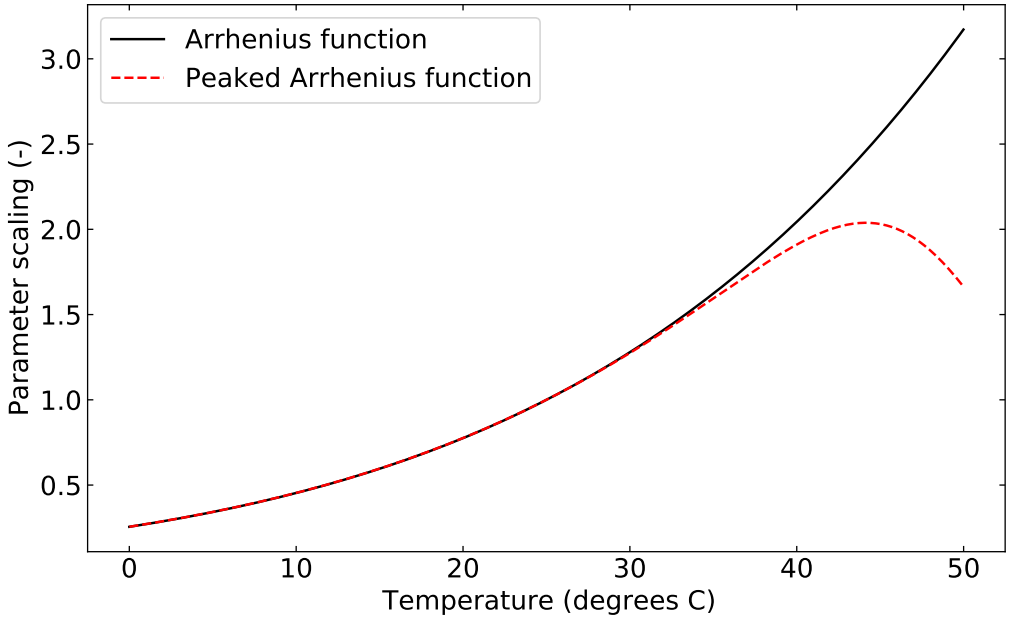


Figure 4.4: The temperature scaling applied to parameters according to the Arrhenius and peaked Arrhenius functions.

Now that we have written down how biochemical model parameters vary with leaf temperature, we need to determine leaf temperature. To model leaf temperature, we use the leaf energy balance as described in [141]. The terms that are described there as contributing to the leaf energy balance are:

- Absorption of solar irradiation
- Absorption of infrared radiation from the surroundings
- Emission of infrared radiation
- Heat conduction and convection
- Heat loss due to evaporation
- Photosynthesis
- Metabolic processes
- Changes in leaf temperature

The contribution of photosynthesis and metabolic processes is typically on the order of a few Watts, while other terms such as the solar irradiation are several hundreds of Watts. So to simplify the model and reduce the number of dependencies between variables we are leaving the photosynthesis and metabolic processes out of the energy balance. Because of the low specific mass of leaves (somewhere in the ballpark of 0.2 $\frac{\text{kg}}{\text{m}^2}$), their specific heat is low and a net energy balance of a few Watts is enough to increase the temperature by a degree Celsius in less than 5 minutes. Because of this, we assume the leaf is in steady-state and the energy balance is zero. This will allow us to calculate the leaf temperature.

The absorption of solar irradiation A_S in $\frac{\text{W}}{\text{m}^2}$ is equal to

$$A_S = a(1+r)I, \tag{4.6.13}$$

where a is the absorptance of the leaf over the whole solar spectrum, r is the fraction of solar irradiation reflected by the surroundings and I in $\frac{\text{W}}{\text{m}^2}$ is the solar irradiation. Assuming there are no nearby objects that reflect a lot of sunlight towards the field in which we are modelling crops and using a symmetry argument (for each photon reflected from another leaf, we can assume the leaf reflects a photon itself), we can set $r = 0$. This yields

$$A_S = aI. \tag{4.6.14}$$

Using the Stefan-Boltzmann law, the absorption of infrared radiation from the environment A_{IR} is equal to

$$A_{IR} = a_{IR}\sigma\left(T_{surr}^4 + T_{sky}^4\right). \tag{4.6.15}$$

Here a_{IR} in $\frac{\text{W}}{\text{m}^2}$ is the infrared radiation absorptance of the leaves, σ in $\frac{\text{W}}{\text{m}^2\,\text{K}^4}$ is the Stefan-Boltzmann constant, T_{surr} in K is the temperature of the surroundings and T_{sky} in K is the effective temperature of the sky. Here we assume that the underside of the leaves absorb infrared radiation coming from the surroundings while the upper side absorbs infrared radiation coming from the sky. Note that T_{sky} is not the actual temperature of the sky, but the temperature a blackbody

that emitted as much radiation as the sky would have. We assume that

$$T_{surr} = T_a, \tag{4.6.16}$$

$$T_{sky} = T_a - 40. \tag{4.6.17}$$

A blackbody with absorptance a will also have emissivity a so the emission of infrared radiation by the leaves, e_{IR} in $\frac{\mathrm{W}}{\mathrm{m}^2}$, is equal to

$$e_{IR} = 2a_{IR}\sigma T_L^4. \tag{4.6.18}$$

The factor 2 in this equation comes from the fact that the leaves emit infrared radiation from both sides. The heat loss due to conduction into the air boundary layer around leaves and then convection away from the leaves, H_c in $\frac{\mathrm{W}}{\mathrm{m}^2}$, is equal to

$$H_c = 2K_{air}(T_a)\frac{T_L - T_a}{\delta_{bl}}, \tag{4.6.19}$$

$$\delta_{bl} = \frac{1}{0.97}\sqrt{\frac{d\nu(T_a)}{v}}. \tag{4.6.20}$$

Here $K_{air}(T_a)$ in $\frac{\mathrm{W}}{\mathrm{m\,K}}$ is the thermal conductivity of the air, δ_{bl} in m the thickness of the boundary layer, d in m the characteristic length of the leaf, $\nu(T_a)$ in $\frac{\mathrm{m}^2}{\mathrm{s}}$ the kinematic viscosity of the air and v in $\frac{\mathrm{m}}{\mathrm{s}}$ the wind speed. Again, the factor 2 is because heat is conducted away from both sides of the leaf. For the range of air temperatures we are concerned with, 273.15 to 323.15 °C, we use, from [141], the following approximations

$$K_{air}(T_a) = 0.0243 + 0.00007(T_a - 273.15), \tag{4.6.21}$$

$$\nu(T_a) = (1.415 + 0.09(T_a - 273.15)) \cdot 10^{-5}. \tag{4.6.22}$$

The heat loss due to transpiration, H_t in $\frac{\mathrm{W}}{\mathrm{m}^2}$, is equal to

$$H_t = J_v H_{vap}(T_a), \tag{4.6.23}$$

where J_v in $\frac{\mathrm{mol}}{\mathrm{m}^2\,\mathrm{s}}$ is the transpiration rate and $H_{vap}(T_a)$ in $\frac{\mathrm{J}}{\mathrm{mol}}$ is the heat of

vaporisation of water, which is equal to

$$H_{vap}(T_a) = 45060 - 425(T_a - 273.15). \tag{4.6.24}$$

Putting this all together, the total energy balance E is equal to

$$\begin{aligned} E =& aI + a_{IR}\sigma\left(T_{surr}^4 + T_{sky}^4\right) - 2a_{IR}\sigma T_L^4 \\ & - 2K_{air}(T_a)\frac{T_L - T_a}{\delta_{bl}} - J_v H_{vap}(T_a) = 0. \end{aligned} \tag{4.6.25}$$

This is a quartic equation in T_L but rather than solving it analytically, which will require us to choose a root, we solve it numerically using the Newton-Raphson method. As our initial guess we take T_a, which should be relatively close to T_L.

4.7 Solar radiation model

To make the simulation more accurate, a solar irradiation model, as described in [130], was implemented. We briefly describe the most important equations here for completeness. All the angles in this section are in degrees, rather than radians.

The solar irradiance at the surface of the earth $I(t)$ in $\frac{\mathrm{W}}{\mathrm{m}^2}$ at a given time t is given by

$$I(t) = \cos(\Theta'(t))\frac{I_0}{R^2(t)}, \tag{4.7.1}$$

where $\Theta'(t)$ is the zenith angle of the sun, corrected for the slope of the field, which is the angle between a line pointing straight up and the sun, I_0 is the solar irradiance at the surface of the earth in $\frac{\mathrm{W}}{\mathrm{m}^2}$ at 1 astronomical unit (AU) and $R(t)$ is the distance between the earth and the sun expressed in AU. $R(t)$ is given by the empirical formula

$$R = 1.00014 - 0.01671\cos(g(t)) - 0.00014\cos(2g(t)), \tag{4.7.2}$$

where $g(t)$ is the mean anomaly, which is a measure of where in its orbit earth is. $g(t)$ is given by

$$g(t) = 357.528 + 0.9856003n(t), \tag{4.7.3}$$

where $n(t)$ is the difference between the current Julian date and the Julian date at 12:00 on the 1st of January 2000. $n(t)$ is equal to

$$n(t) = 2432916.5 + 365(y - 1949) + \lfloor y - 1949 \rfloor + d(t) - 2451545, \tag{4.7.4}$$

where y is the year in which we started the simulation and $d(t)$ is the current day (equal to the start day of the simulation + t), including the fractional part of the day. $\cos(\Theta'(t))$, the zenith angle corrected for the slope of the field is given by

$$\cos(\Theta') = \cos(\Theta(t))\cos(s) + \sin(\Theta(t))\sin(s)\cos(\alpha(t) - saz), \tag{4.7.5}$$

where $\Theta(t)$ is the zenith angle (for a flat field), $\alpha(t)$ is the right ascension of the sun (the celestial equivalent of longitude), s is the slope of the field and saz is the surface azimuth relative to north. $\alpha(t)$ is given by

$$\tan(\alpha(t)) = \cos(ep(t))\frac{\sin(l(t))}{\cos(l(t))}, \tag{4.7.6}$$

where $ep(t)$ is the obliquity of the ecliptic, the angle between the rotational and orbital axis of earth and $l(t)$ is the ecliptic longitude, the angular distance of the sun along the ecliptic. These two quantities are given by

$$ep(t) = 23.439 - 0.0000004n(t), \tag{4.7.7}$$

$$l(t) = 280.460 + 0.9856474n(t) + 1.915\sin(g(t)) + 0.20\sin(2g(t)). \tag{4.7.8}$$

$\Theta(t)$ is given by

$$\cos(\Theta(t)) = \sin(\delta(t))\sin(lat) + \cos(\delta(t))\cos(lat)\cos(h(t)). \tag{4.7.9}$$

where $\delta(t)$ is the declination, the celestial equivalent of latitude, of the sun, lat is the latitude and $h(t)$ is the hour angle. These quantities are given by

$$\sin(\delta(t)) = \sin(ep(t))\sin(l(t)), \tag{4.7.10}$$

$$h(t) = 6.697375 + 0.0657098242n(t) + 24(t - \lfloor t \rfloor), \tag{4.7.11}$$

where $t - \lfloor t \rfloor$ is the fractional part of t, the local time.

4.8 Final Implementation Details

After implementing all these models, a number of issues still had to be resolved. Some of these are code optimisations needed to cut down computational time and memory requirements, which were increased greatly by the numerical root finder and some differences in model assumptions.

First of all, the implementation of the Penman-Monteith equations that govern soil evaporation and crop transpiration was not performing as expected: When testing the C3 photosynthesis model in rice, the simulated evaporation values differed from field measurements by more than one order of magnitude. Upon reviewing the code, there were a number of assumptions which required review. We will quickly detail the changes made here.

For the crop evaporation rate, J_v in $\frac{\text{mol}}{\text{m}^2\,\text{s}}$, we use equation 13 on page 210 of [134], which states that

$$J_v = \frac{\Delta I_s + \rho C \frac{VPD}{r_a}}{\lambda(\Delta + \gamma(1 + \frac{r_s}{r_a}))} \tag{4.8.1}$$

where Δ in $\frac{\text{hPa}}{\text{K}}$ is the vapour pressure slope, I_s in $\frac{\text{W}}{\text{m}^2}$ the solar irradiation rate, ρ in $\frac{\text{kg}}{\text{m}^3}$ the density of air, C in $\frac{\text{J}}{\text{kg K}}$ the specific heat capacity of air, VPD in hPa the vapour pressure deficit, r_a in $\frac{\text{s}}{\text{m}}$ the aerodynamic resistance, λ in $\frac{\text{J}}{\text{mol}}$ the latent heat of water vaporisation, $\gamma = \frac{CP}{\lambda MW_{ratio}}$ in $\frac{\text{hPa}}{\text{K}}$ the psychrometric constant, P in hPa is the air pressure, MW_{ratio} is the ratio between the molecular weights of water vapour and air and r_s in $\frac{\text{s}}{\text{m}}$ is the stomatal resistance. For the soil evaporation E_s in $\frac{\text{mol}}{\text{m}^2\,\text{s}}$ we use equation 8 on page 208 of [134], which states that

$$E_s = \frac{\Delta I_{soil} + \rho C \frac{VPD}{r_a}}{\lambda(\Delta + \gamma)} \tag{4.8.2}$$

where I_{soil} in $\frac{\text{W}}{\text{m}^2}$ is the solar irradiation rate on the soil surface. Previously, these values were then modified as follows:

$$J_v^{actual} = sJ_v \tag{4.8.3}$$

$$E_s^{actual} = (1 - s)E_s \tag{4.8.4}$$

with $s = 1 - e^{-k \cdot LAI}$, as in equation 4.1.2, the same scaling as used by the Lintul photosynthesis model [158, 195]. However, the terms depending on the vapour pressure deficit in equations 4.8.1 and 4.8.2 model the transpiration/evaporation due to the difference in hydraulic potential between the leaves/soil and the air and is independent on the amount of irradiation hitting the soil (otherwise the radiation would be included as a factor). Instead, I_{soil} should take into account the interception of solar radiation by the leaves, which we did by introducing a multiplier based on the leaf area index. The soil radiation is then calculated by the same model that calculates the radiation intensity hitting the leaves, scaled appropriately by this multiplier. Furthermore, since we are not using the Lintul model, we set $s = LAI$, the leaf area index because this is the scaling we use for the light-limited photosynthesis rate and the leaf temperature. In order to ensure backwards compatibility we add a switch to choose between this scaling or the previous behaviour with the default being the previous behaviour.

Another difference in model assumptions concerns water flow out of roots. If solving the equation governing xylem pressure led to water flowing out of the roots, the water that moved out of the roots was not added to the soil because outflow was not modeled. This was because outflow only happens when the soil is (locally) drier than the roots, which was not relevant until the drought-related models and the day-night cycle irradiation model described in this chapter were added. We altered the soil water model to accept water efflux from roots, which means OpenSimRoot can now simulate hydraulic lift, a process that is important for the daily changes in soil water content [30, 169].

Test simulations with very dry soils saw frequent crashes because the water model generated "NaN" (not a number) values. This was traced back to very low hydraulic conductivity values generating positive water hydraulic potentials. Positive hydraulic potentials correspond to higher soil water content than are possible at full saturation and generate NaN values when entered into the Van Genuchten equations [69], which OpenSimRoot uses to calculate soil water retention curves and hydraulic conductivity. These equations have asymptotic behaviour near the residual and saturated hydraulic potentials, where they are not seen as suitable [120]. Updating the OpenSimRoot implementation of the Van Genuchten equa-

tions to better model soil water flows at very high or low hydraulic potentials is desirable but beyond the scope of this chapter. For now, in order to prevent crashes in a backwards compatible manner, we added an option to set a minimum soil hydraulic conductivity value, any values lower than this will be set to this minimum value. This prevents numerical issues if needed, but if it is not set or the threshold is not reached, the model functions just as previously. This change greatly improved the stability of the soil water simulation, which was the cause of a majority of crashes before this.

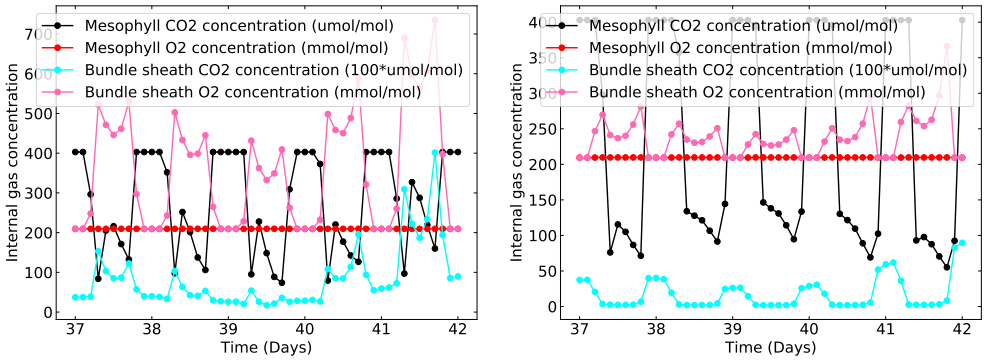
After initial testing we noted that the water uptake values as calculated by the Doussan model did not match the transpiration prescribed by the shoot, which should be the case. One source of inaccuracy was that the Doussan model was used to calculate the water uptake rates for each root segment, which were integrated. Instead of this, we added an option for the Doussan model to calculate the total water uptake (so not the rate, but the integration of this over time) of root segments and write them to appropriate tables directly. This, in combination with a fixed global timestep greatly improved the accuracy.

The numerical root finder caused memory use and computation time needed for simulations to increase dramatically, especially when simulating C4 photosynthesis. Because there is a large number of interrelated equations for C4 photosynthesis (equations 4.5.11, 4.5.12, 4.5.13, 4.5.14, 4.5.10, 4.5.1, 4.5.2, 4.5.3 and 4.1.13, 4.8.1, 4.6.5), the numerical root finder typically has to iterate hundreds or even thousands of times to converge to a simultaneous solution. This in itself is not computationally expensive but interactions with other parts of the OpenSimRoot engine increased the computational load beyond reasonable limits. First of all, it is often the case that multiple variables that depend on each other have to be integrated forward in time (e.g. leaf area and photosynthesis rate). OpenSimRoot tracks these internally and marks them as 'predicted' values, with the integration repeated for all 'predicted' values. However, when the root finder is iterating it can generate upwards of millions of 'predicted' values by repeatedly requesting, for example, leaf area. This caused a small simulation which should only use about 400MB of memory to use more than 4GB of memory. This issue was addressed by adding a cache for values which is used when the root finder

is iterating, cutting down on both memory use and computation time. A second cause for increased computational load was that the code for several state variables (solar irradiation, leaf temperature, anything with variables varying by leaf temperature) was relatively computationally expensive to evaluate and because this code was evaluated hundreds of times per timestep, it added to the computational load disproportionately. Since the solar irradiation depends only on time and input variables, this code was optimised by creating an internal cache where values for each time could be saved. Likewise, the code for any state variable with temperature-dependent parameters was optimised by saving the leaf temperature and the corresponding parameters in a cache so that they only have to be recalculated if the leaf temperature changes. With some other small optimisations, the code slowdown due to the numerical root finder and C4 models was reduced to about a factor 2.

4.9 Simulation results

With everything implemented and working properly, we are now able to generate outputs with OpenSimRoot. Simulating a maize plant in well-watered and drought conditions for 42 days we can see the effect of water availability on key model state variables. Figures 4.5a and 4.5b show the CO_2 and O_2 concentrations in the mesophyll and bundle sheath cells under well-watered and drought conditions during the final 5 days of the simulation. Note that the unit for the mesophyll CO_2 concentration differs from the units for the other concentrations in order to improve readability. During the day the bundle sheath and mesophyll CO_2 concentrations reduce because CO_2 gets fixed by photosynthesis, while the O_2 concentration increases, being a byproduct of photosynthesis. The variation in mesophyll O_2 concentration is too small to see in this figure because it rapidly equilibriates with the atmospheric concentration. The reason this does not happen for the bundle sheath O_2 concentration is that the bundle sheath conductance is many orders of magnitude smaller than the stomatal conductance. Note also that the bundle sheath CO_2 and O_2 concentrations are far lower in the drought-stressed plant than in the well-watered plant, because the stomatal aperture is lower.



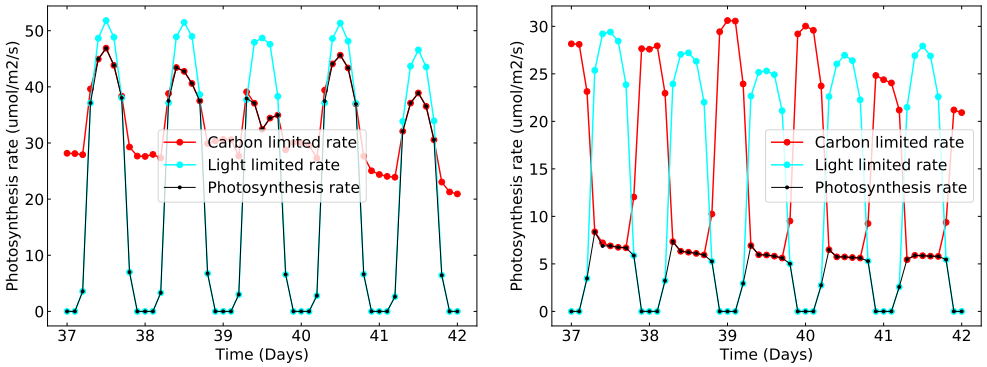
(a) Internal gas concentrations under well-watered conditions.

(b) Internal gas concentrations under drought conditions.

Figure 4.5: Simulated internal gas concentrations during the last 5 days of 42-day OpenSimRoot simulated maize plants under well-watered and drought conditions. Note that the unit for the mesophyll CO_2 concentration is $\frac{\mu mol}{mol}$, the unit for the bundle sheath CO_2 concentration is 100 $\frac{\mu mol}{mol}$ while for the two O_2 concentrations it is $\frac{mmol}{mol}$.

Figures 4.6a and 4.6b show the carbon-and light-limited assimilation rates during the final 5 days of the simulation for well-watered and drought conditions respectively. The light-limited rates mirror the daily variation in solar radiation, as expected, while the carbon-limited rates vary as well but remain nonzero during the night. Under well-watered conditions we see the carbon-limited rate increasing during the day because of stomatal opening while under drought conditions the carbon-limited rates decrease during the day. This is because photosynthesis is carbon-limited so most of the carbon coming into the leaves is assimilated, leading to low equilibrium concentrations.

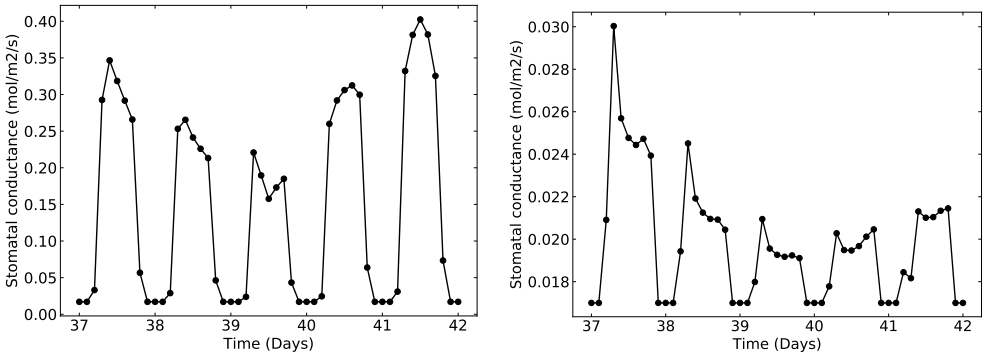
Figures 4.7a and 4.7b show the stomatal conductances during the final 5 days of the simulation under well-watered and drought conditions. The first thing to note is that under drought, stomatal conductances are more than 10 times lower during the day as compared to well-watered conditions. There is also quite a bit of daily variation in both cases. In the well-watered case this is due to differences in leaf temperature and atmospheric condictions as well as slight differences in water stress levels (the plant is experiencing very minor water stress levels during the day even in these well-watered conditions). For the drought conditions this is also the case but the water stress level explains a larger part of the variation and is increasing over time.



(a) Assimilation rates under well-watered conditions.

(b) Assimilation rates under drought conditions.

Figure 4.6: Simulated carbon-limited and light-limited photosynthetic assimilation rates during the last 5 days of 42-day OpenSimRoot simulated maize plants under well-watered and drought conditions. The actual assimilation rate at any given time is the minimum of these two rates.



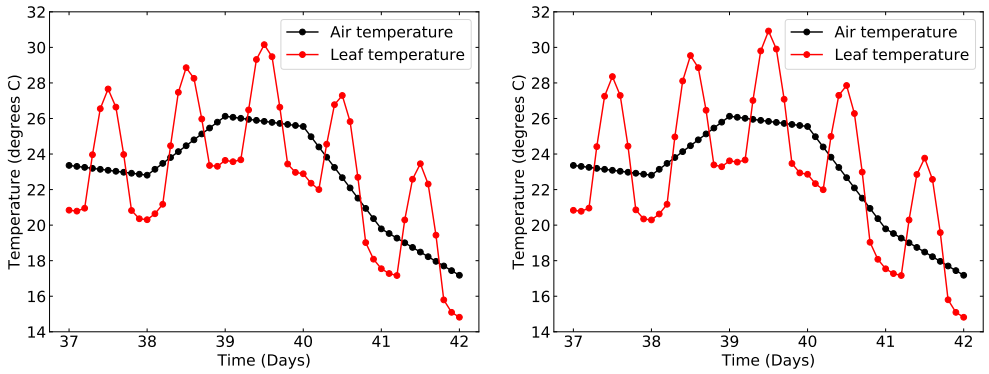
(a) Stomatal conductance under well-watered conditions.

(b) Stomatal conductance under drought conditions.

Figure 4.7: Simulated stomatal conductances during the last 5 days of 42-day OpenSimRoot simulated maize plants under well-watered and drought conditions.

Figures 4.8a and 4.8b show the air and leaf temperature during the final 5 days of the simulation. We clearly see the daily variation in temperature and note that the difference between the well-watered and drought cases is very small.

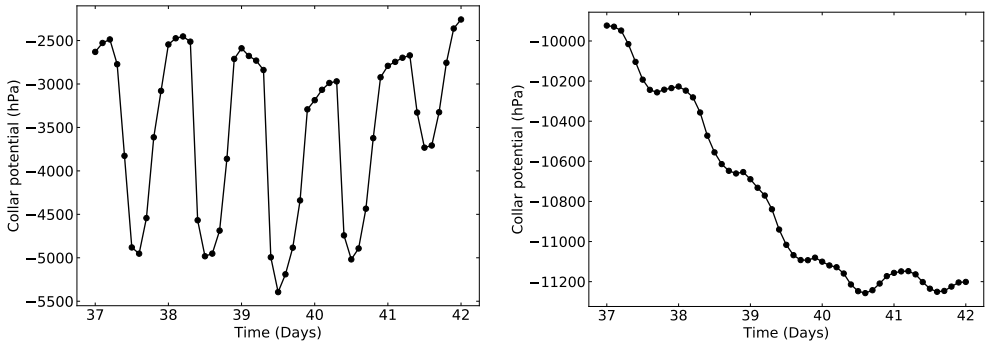
Finally, figures 4.9a and 4.9b show the collar water potentials during the final 5 days of the simulation. Since transpiration rates are very low during the night, the collar potentials are a lot higher then. For the well-watered case, the collar potential hovers around -3000 hPa during the night, spiking down to -5000 hPa during the day. For the drought-stressed plant on the other hand, we see a steady decline evening out around -11200 hPa.



(a) Air and leaf temperatures under well-watered conditions.

(b) Air and leaf temperatures under drought conditions.

Figure 4.8: Simulated air and leaf temperatures during the last 5 days of 42-day OpenSimRoot simulated maize plants under well-watered and drought conditions. The actual assimilation rate at any given time is the minimum of these two rates.



(a) Collar potentials under well-watered conditions.

(b) Collar potentials under drought conditions.

Figure 4.9: Simulated hydraulic water collar potentials during the last 5 days of 42-day OpenSimRoot simulated maize plants under well-watered and drought conditions. The actual assimilation rate at any given time is the minimum of these two rates.

4.10 Application of the new capabilities

We use the C4 photosynthesis and drought response model described in this chapter to test the hypothesis put forward in [112] that a *steep, cheap and deep*, or parsimonious, phenotype performs well under drought. To do this, we simulate two different maize phenotypes for 52 days at three different planting densities in three different scenarios: well-watered, drought and topsoil drought. The differences between the two phenotypes are summarised in Table 4.1. An Iowa soil was simulated, more specifically the Iowa BOOI4 site with latitude 42.02094 and longitude -93.7743. The weather data for this site was also used and plants were simulated from the 1st of May 2015. Figure 4.10 shows the initial distribution

of water in the soil for the three different environments. There was precipitation in the well-watered scenario but not in the two drought scenarios. The plants were simulated at planting densities of 5, 7.4 and 10 $\frac{\text{plants}}{\text{m}^2}$. For each of these 18 different scenarios we do 10 repetitions.

Phene	Reference phenotype	Parsimonious phenotype	Unit
Seminal root number	6	3	-
Nodal root number	54	27	-
Branching density	4	2	$\frac{\text{branches}}{\text{cm}}$
Axial branching angle	45	15	degrees

Table 4.1: The phene values for the reference and *steep, cheap and deep* phenotypes. For branching density and axial branching angle different root classes have different values, a representative value was chosen. The branching angle is with respect to the down direction.

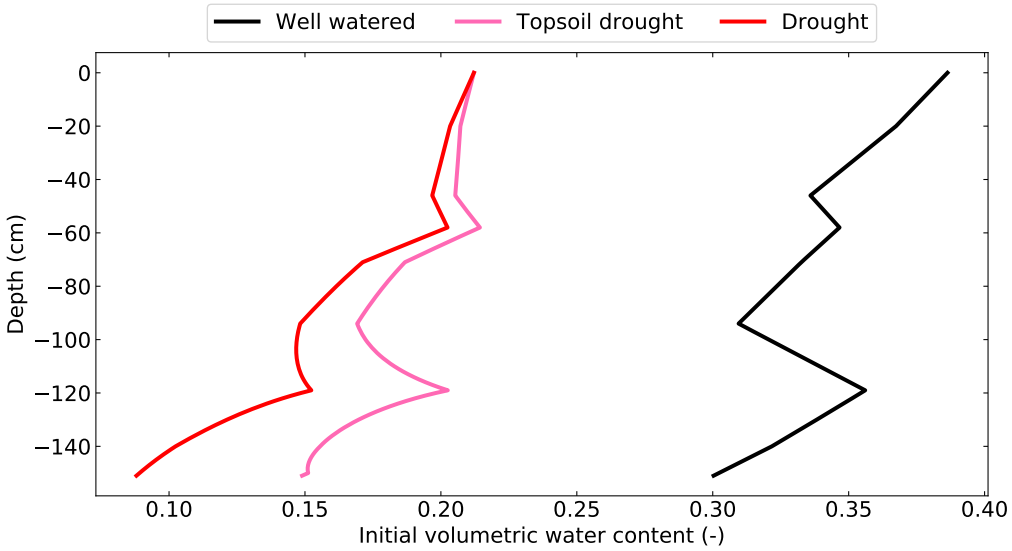


Figure 4.10: Initial soil water distribution for the three different environments. The sharp breaks are due to the sharp borders between different soil bands.

We test the hypothesis that the *steep, cheap and deep* phenotype will perform better under drought conditions, by which we mean that it will correspond to higher yield. Because we can not simulate yields in OpenSimRoot (maize is not parametrised up to flowering, let alone grain filling), we will use vegetative growth, in other words, shoot biomass as a proxy for yield. Figures 4.11 and 4.12

show the relative and absolute root length distributions in the soil for medium planting densities. We see that the *steep, cheap and deep* phenotype does indeed have far less root length overall, while it is concentrated more in the deep soil.

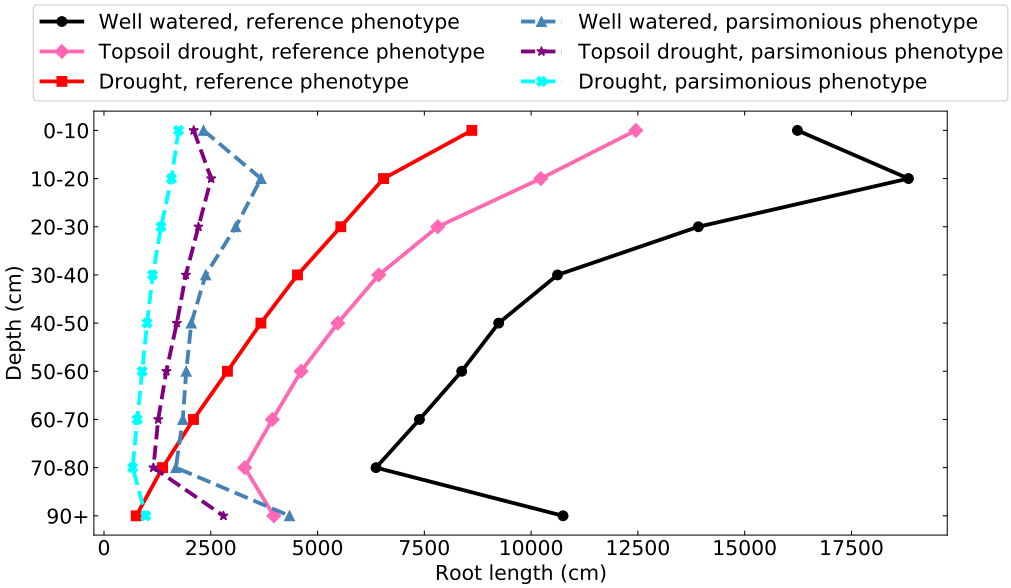


Figure 4.11: Distributions of root length depth at medium planting density.

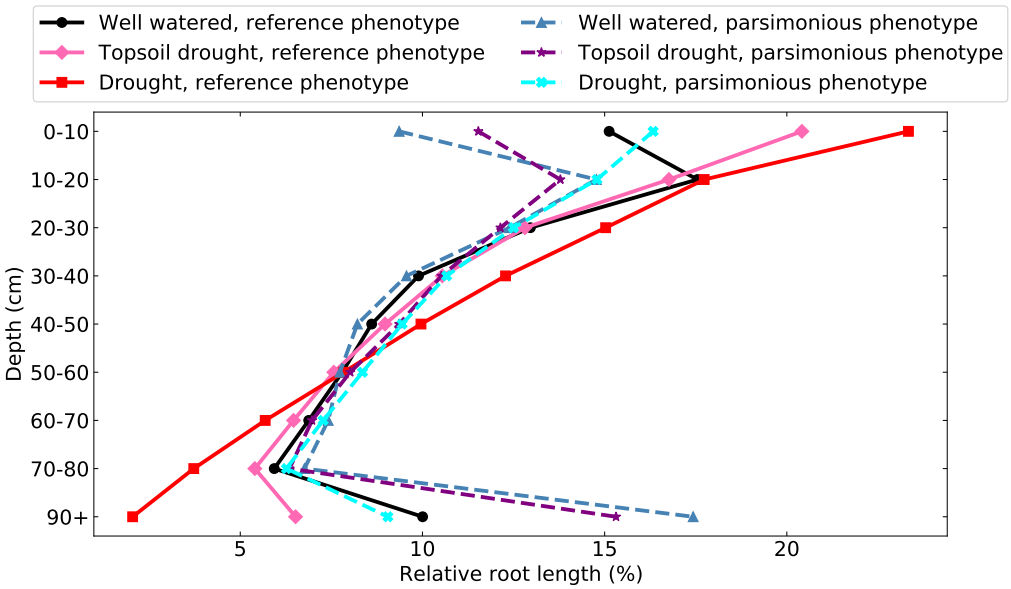


Figure 4.12: Distributions of relative root length depth at medium planting density. For each repetition, the root length within each interval is normalised with respect to the total root length and then these normalised distributions are averaged.

Figure 4.13 shows that while both phenotypes have similar shoot dry weight in the well-watered scenario at the lower 2 planting densities, the *steep, cheap and deep*, or parsimonious, phenotype has an $8g$ greater shoot dry weight compared to the reference phenotype at a planting density of 10 $\frac{\text{plants}}{\text{m}^2}$, and increase of about 20%. In the topsoil drought scenario, the *steep, cheap and deep* phenotype has a 5 g to almost 9 g greater shoot dry weight, which represents an increase of 20 to 25%. In the complete drought scenario, the difference is 2.3 g to almost 4 g or about 20%. This suggests that a more parsimonious phenotype does perform better when water (and therefore carbon) is limiting, which also happens at the greatest planting density in the well-watered scenario. This last point is important because maize in the USA is planted at increasingly greater densities, increasing with about 0.07 $\frac{\text{plants}}{\text{m}^2.\text{y}}$ [7]. The greatest planting density in these simulations is in line with current commercial planting densities in the USA.

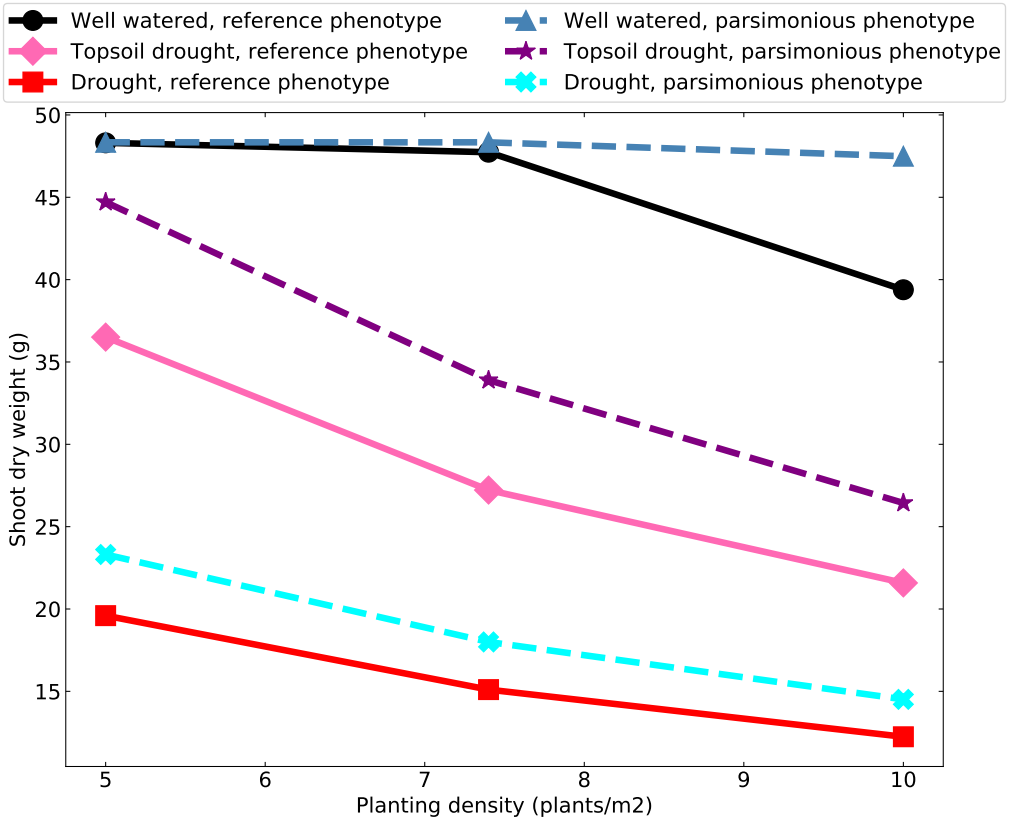


Figure 4.13: Mean shoot dry weight for al 18 scenarios.

However we see from Figure 4.14 that the parsimonious phenotype takes up less water from the soil than the reference phenotype in all three scenarios, though the difference is relatively small. Total carbon assimilation is similar, with the reference phenotype having greater carbon assimilation in most cases, as shown by Figure 4.15. This means that the parsimonious phenotype is not necessary able to access more water than the reference phenotype.

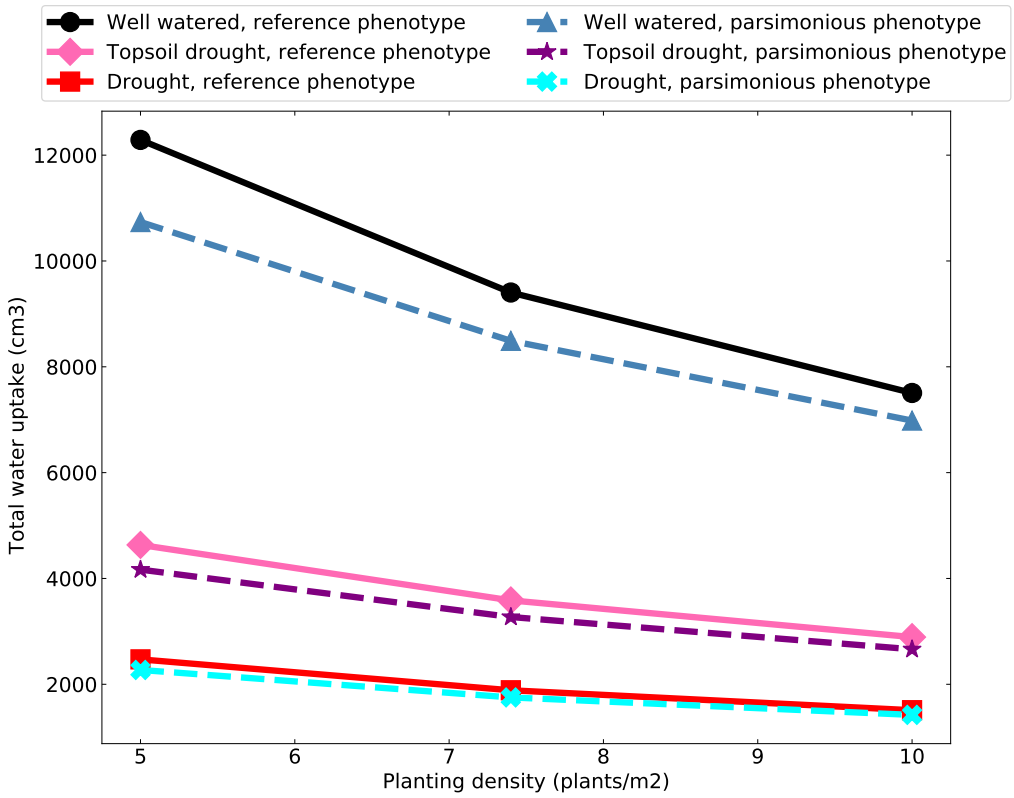


Figure 4.14: Total water uptake for all 18 scenarios.

One of the advantages of simulations is that we have access to information that would be very difficult to obtain for real plants, such as the carbon budgets and the total costs of root maintenance. We see from Figure 4.16 that the parsimonious phenotype spends significantly less carbon on root maintenance (this includes respiration and exudation) than the reference phenotype. Figure 4.17 shows that this is also true in terms relative to the total carbon assimilation. Of course, this is not surprising, since we already saw in Figure 4.11 that the parsimonious phenotype has a much smaller root system than the reference phenotype.

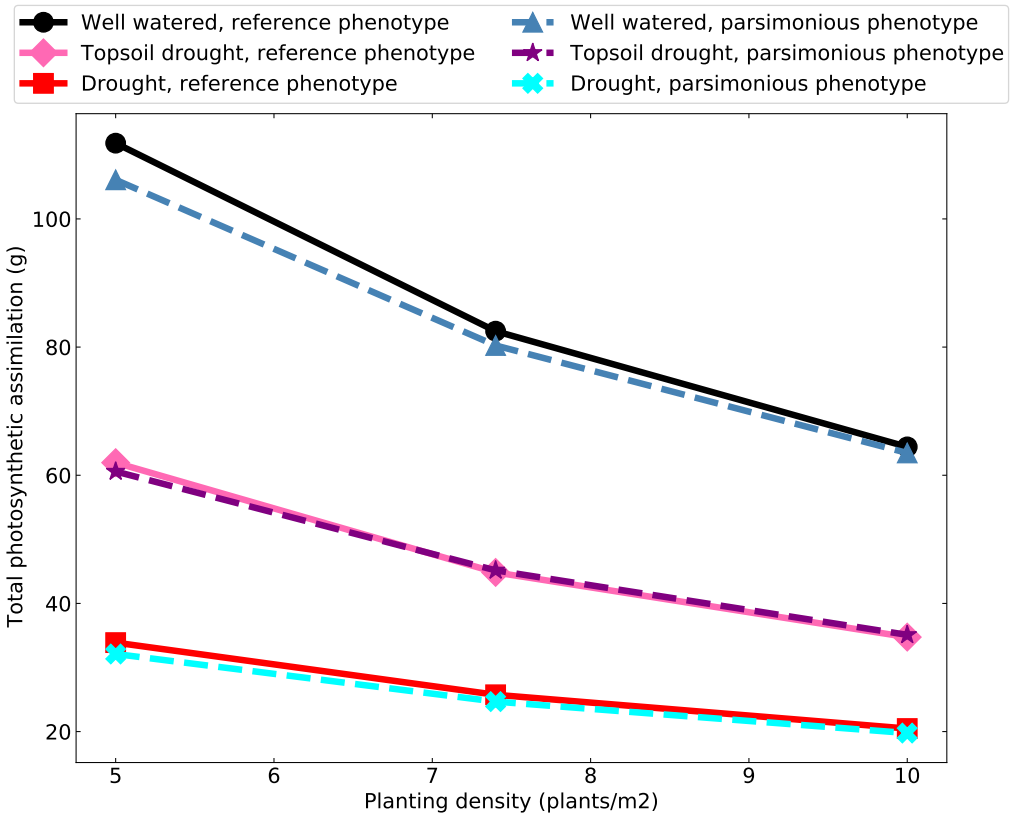


Figure 4.15: Total photosynthetic carbon assimilation for all 18 scenarios.

We conclude that a more parsimonious root system phenotype does indeed provide advantages under drought. In the scenario where initial water content was reduced in all soil layers, this was due to the fact that the parsimonious phenotype saw a significant reduction in carbon costs for root respiration and exudation while total water uptake and carbon assimilation were only slightly smaller. In other words, the parsimonious phenotype was more carbon efficient than the reference phenotype, investing less carbon in roots and thus being able to grow a bigger shoot. Because we have complete information, we can quantify this. As it turns out, the parsimonious phenotype is able to take up more water per gram of carbon invested in the root system (this includes both biomass as well as maintenance costs in the form of respiration and exudation), as shown in Figure 4.18. The parsimonious phenotype is about twice as efficient in taking up water, taking up more than 0.4 $\frac{\mathrm{L}}{\mathrm{g}}$ in the drought scenarios where the reference phenotype takes up around 0.2 $\frac{\mathrm{L}}{\mathrm{g}}$. In the well-watered scenario the difference is a factor 2.5 to 3, depending on planting density, in favour of the parsimonious phenotype. On the other side of the equation, as we see in Figure 4.19 the amount of total biomass produced per litre water taken up is greater for the parsimonious phenotype as

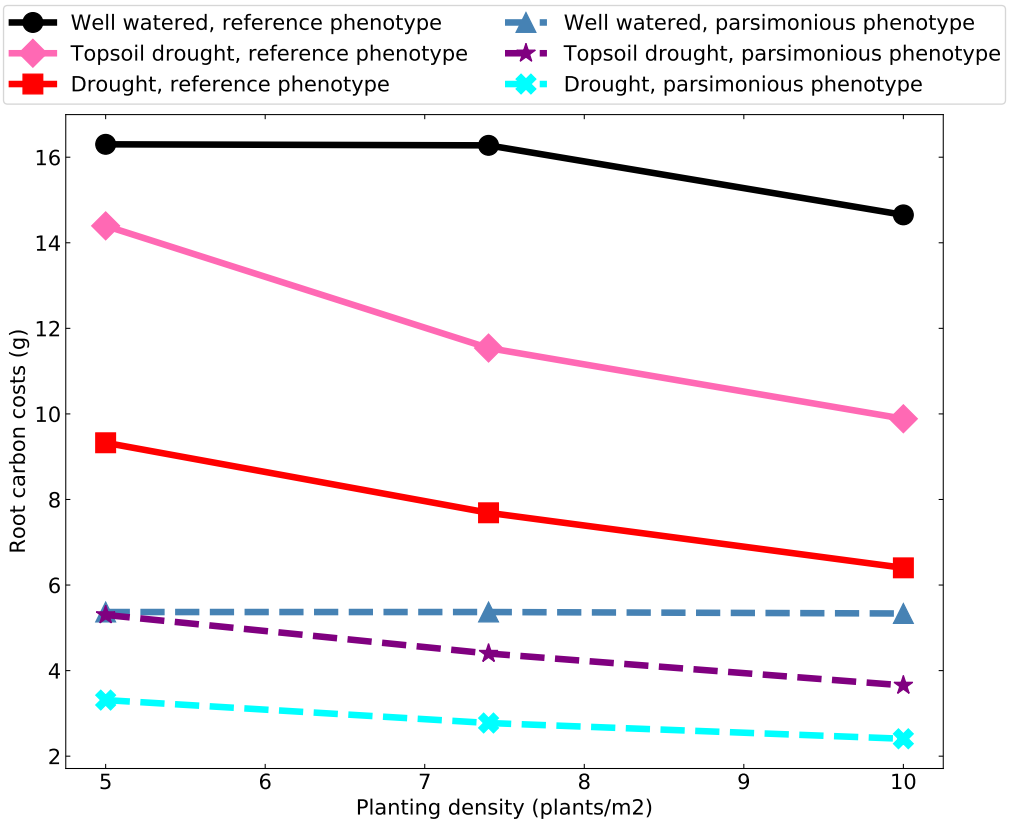


Figure 4.16: Root maintenance costs. This is the root respiration plus the carbon costs of root exudation.

well, though by a smaller margin.

Figure 4.20 shows that the amount of shoot biomass produced per litre of water taken up is greater for the parsimonous phenotype across planting densities and environments. This is partly due to the fact that the parsimonious phenotype produces a lot less root biomass and a lot more shoot biomass. However, we also see that the amount of shoot biomass produced is greater in the drought scenarios as compared to the well-watered scenario. This implies that plants increase their water use efficiency under drought by reducing stomatal conductance, which makes sense since adjusting stomatal conductance is an evolutionary adaptation to drought. This provides an explanation for what we saw in Figure 4.19: the parsimonious phenotype is more efficient in producing biomass because it is slightly more drought stressed during the simulation (data not shown). Because of the increased drought stress, the *steep, cheap and deep* phenotype reduces transpiration and increases its water use efficiency.

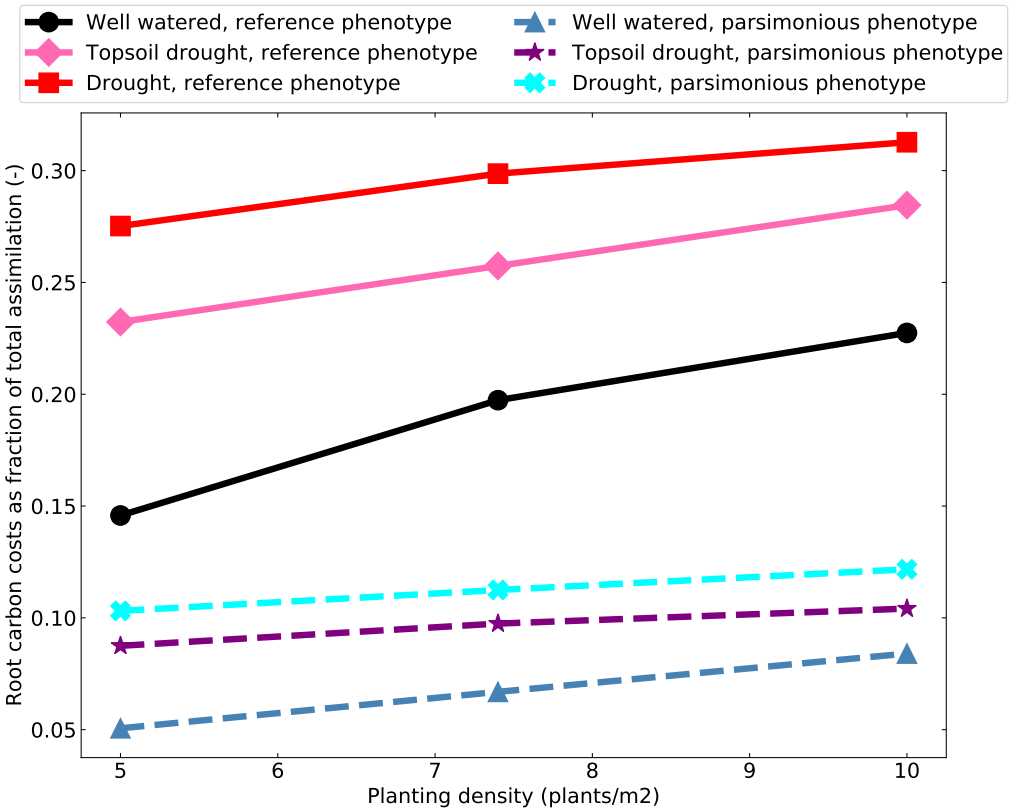


Figure 4.17: Root carbon maintenance costs relative to gross total carbon assimilation.

We conclude that a *steep, cheap and deep* phenotype is better suited to conditions of limited water availability in large part because of the 'cheap' part. Total water uptake and photosynthetic assimilation are slightly greater for the reference phenotype but because the *steep, cheap and deep* phenotype spends less carbon on root biomass, respiration and exudation, it is able to spend limited resources more efficiently. This means it is able to grow a bigger shoot. Of course, a bigger shoot also means higher potential transpiration rates so in the case that the drought continues for very long, this could mean that the parsimonious phenotype reaches wilting point sooner than the reference phenotype. Another possible scenario is that water availability through precipitation is very intermittent. In a sandy soil where water drains away quickly, a larger root system would be able to take up more water than a parsimonious phenotype before it drains away. When the root phenotypes of neighbouring plants differ, we expect the advantage of a parsimonious root phenotype to diminish, since a neighbour with a more extensive root system could potentially take up the limited supply of water faster. More research, both through simulations and in the field, is needed to get a better

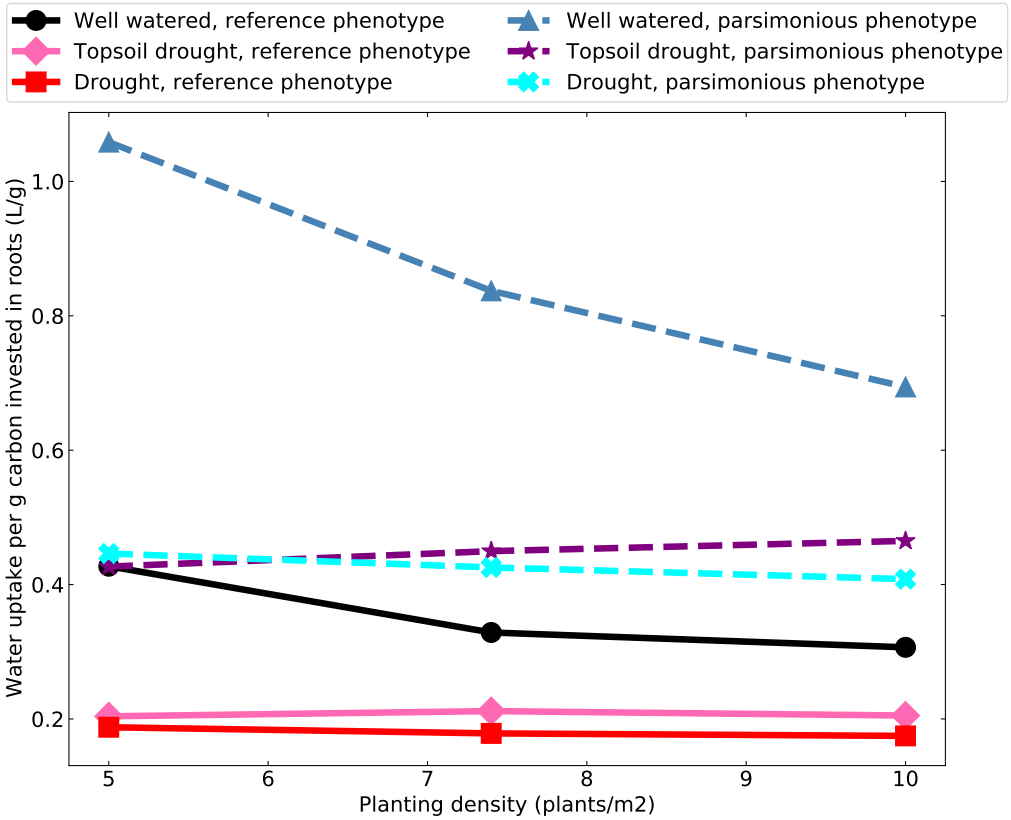


Figure 4.18: The amount of water taken up per gram of carbon invested in the root system. This includes the carbon invested in biomass as well as the maintenance costs of respiration and exudation.

understanding of the interaction between specific drought scenarios, soil type and root system phenotype.

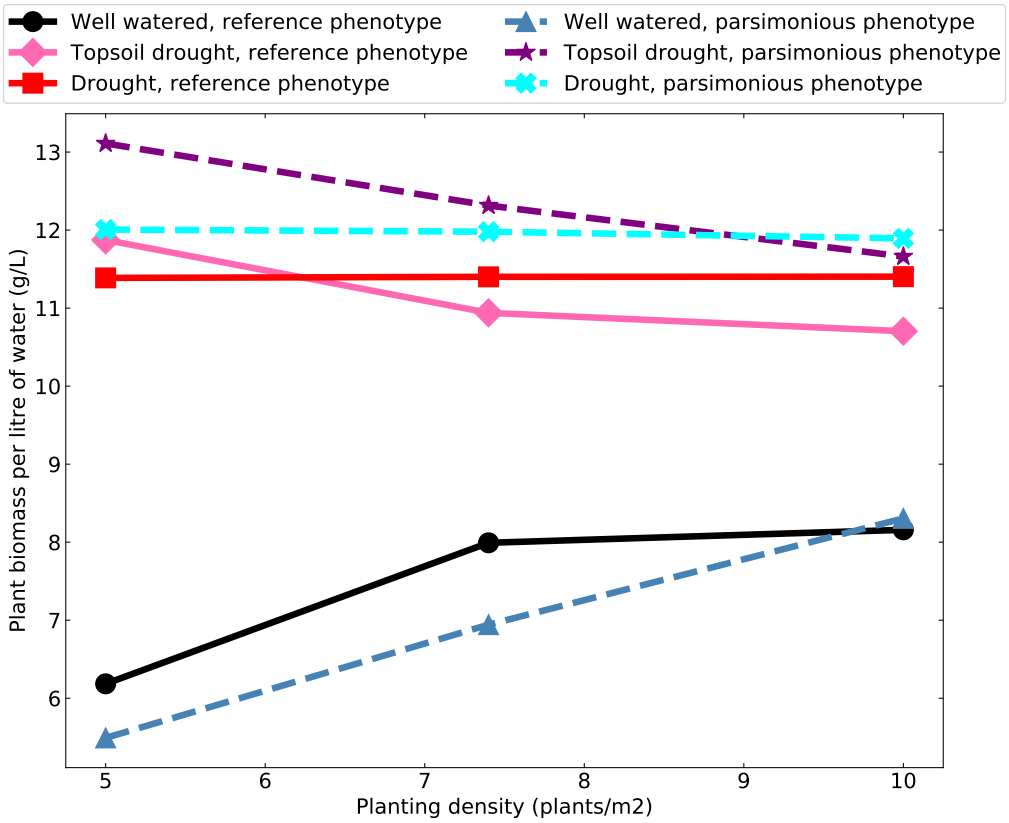


Figure 4.19: The amount of total biomass produced per litre of water taken up.

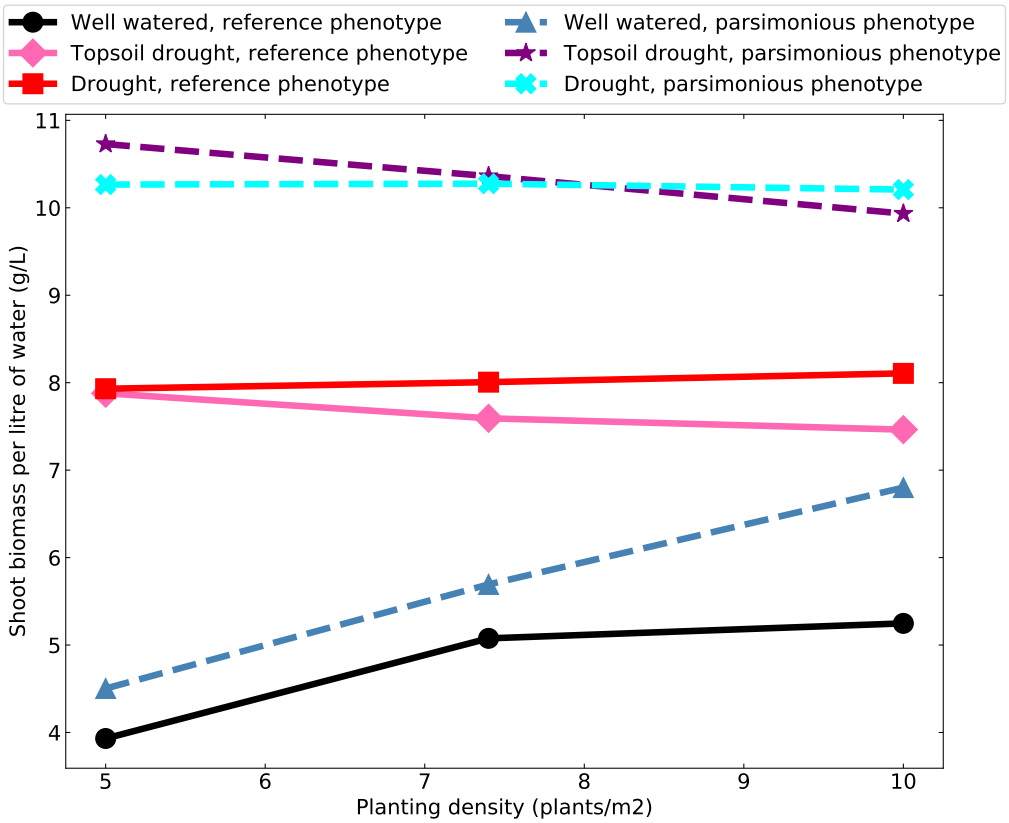


Figure 4.20: The amount of shoot biomass produced per litre of water taken up.

Chapter 5

Optimising Root System Architecture using Emulation

The research in this chapter was done in collaboration with doctor Ian Vernon (Department of Mathematical Sciences, Durham University) to whom we are very grateful. Without his guidance and knowledge this would not have been possible.

OpenSimRoot, by virtue of incorporating a wide range of different models, is relatively computationally expensive, with simulations typically running for 2 to 20 hours on modern processors (e.g. Intel Xeon Gold 6138 20C 2.0GHz CPU). Since OpenSimRoot is not able to take advantage of parallel processing, running OpenSimRoot on a high-end system does not decrease the time needed by much, though a high performance computing cluster is typically able to run many OpenSimRoot simulations in parallel. For root systems that have large numbers of roots, such as rice, runtime can even be as long as several days. Memory usage is usually between 500MB and 12GB but can, again, be larger than this.

Typically, OpenSimRoot is used to explore the interactions between several root properties and a number of different environmental factors such as soil nutrient availability in a factorial design where each combination of possible inputs is used. In such a setup, the number of simulations required grows rapidly as more parameters are varied. Suppose for example that there are two different values for each parameter. With just 10 different parameters this would already yield 1024 different combinations of inputs. Because there is a random component to root elongation rates, root branching and root growth direction, we usually want

to repeat a simulation a number of times with different random number generator seeds, increasing the number of simulations needed. For root properties such as lateral branching density, simulating a range of values is usually preferable to simulating two extremes in order to get a better idea of the effects on plant development. The large number of parameters OpenSimRoot requires in combination with the computational costs of a simulation mean that the scope of simulation studies with OpenSimRoot is limited, unless we use mathematical techniques to decrease the amount of computational resources needed. In this chapter we will lay out an approach to overcome these limitations.

We will seek to maximise the shoot dry weight of maize subjected to nitrogen, phosphorus and potassium stress over a large number of root traits. Because phenotypes with high root length density in the top soil perform well under phosphorus limitations, while parsimonious (frugal, sparse) phenotypes with high rooting depth perform well under nitrogen limitations, if both nutrients are limiting a plant will probably need elements of both these contrasting architectures in order to perform well. Since there are many ways to combine these contrasting phenotypes, it is possible there are a number of distinct combinations that perform well under these two stresses. Since we are not just interested in the best performing root architectures, but also in synergies between phenes and different phene combinations that lead to high shoot biomass, we don't simply want to find the set of inputs that leads to the highest shoot biomass but want to know about the phenotypes whose biomass is in the top 5%.

Doing this naively would take hundreds of thousands or even millions of hours of computing time, which would take several years without access to high performance computing facilities and would exceed the computing budget available to us. To reduce computing time, we will use emulation techniques to rapidly search parameter space. Concretely, this means we will construct a simplified model that can approximate OpenSimRoot outputs at a fraction of the computational cost and use this to determine which regions of parameter space are most likely to yield good results. As an example of an emulator, after running an initial set of OpenSimRoot simulations we could fit a linear regression model which maps the relevant inputs to outputs. Then instead of running OpenSimRoot for several

hours to see how inputs translate to outputs, we would only need to multiply a vector with a matrix. This would probably not yield very good predictions, since it's unlikely OpenSimRoot maps inputs to outputs linearly, which is why we need to put a little more thought in the construction of our emulator.

In order to find the root parameters that maximise shoot biomass in a given environment, we adapt a technique known as Bayesian history matching. This technique originated in the oil industry, where it is used to calibrate oil reservoir models so that their predictions match up with historical records (hence the name) [40]. Since then it has been applied in a variety of contexts such as galaxy formation [196] to disease modelling [6], Arabidopsis hormonal crosstalk modelling [197], climate modelling [210] and traffic simulation [20]. Note that in the examples mentioned, this technique was used to find parameters that make models match observations, which means minimising an error measure or maximising a likelihood. Instead of minimising an error function though, we want to maximise a model output.

In short, the idea behind Bayesian history matching is to rule out regions of parameter space which are 'implausible' to satisfy our demands, in our case we will rule out the regions of parameter space that are unlikely to have high shoot biomass. This is done with the help of an emulator, which in our case not only provides an expected value for any given inputs, but also provides information on the uncertainty associated with this prediction. With this information we can estimate the probability that an output falls in a given range, which will be used to determine if we include a point or not. By running successive 'waves' of OpenSimRoot simulations and constructing different emulators based on each of these, we can zoom in on regions of interest in parameter space with increasing accuracy and ever higher shoot biomasses.

The emulators we will use consist of three parts: A regression in low-order polynomials, a Gaussian process and a noise term. The regression in low-order polynomials is meant to capture the large-scale behaviour of OpenSimRoot in parameter space. The Gaussian process aims to capture fluctuations and interactions on smaller scales and adds an uncertainty estimate to every prediction. The noise

term is there because our emulator is an imperfect approximation of OpenSimRoot (which is an imperfect model of reality). The noise term also accounts for the fact that OpenSimRoot is not strictly deterministic but has stochastic elements. This includes root growth directions, branching rates and root elongation rates, all of which have stochastic components. We will do 10 repetitions for each set of parameters to account for this but this still leaves us with an imperfect estimate of the sample mean, which the noise term accounts for.

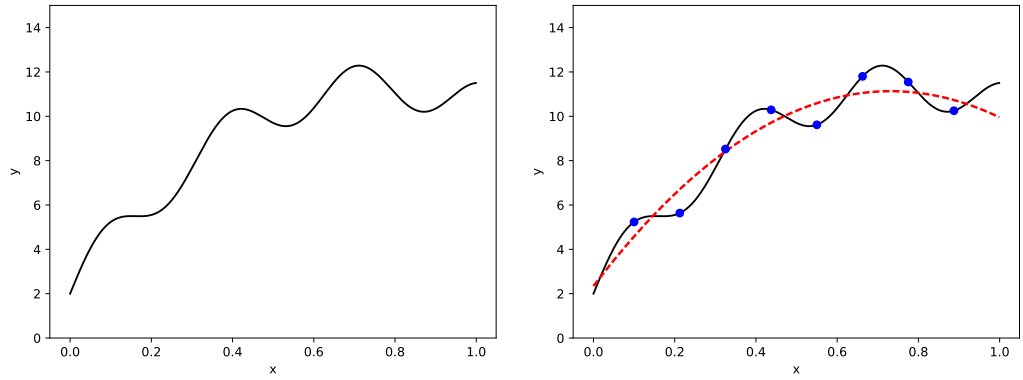
5.1 Illustrative example

To illustrate some important features of Gaussian processes, we apply the procedure described in general terms above to a 1-dimensional function. All the steps will be explained briefly, more thorough explanations of all the steps as well as a number of important definitions will be provided in sections 5.2 and 5.3. The function we use for the purpose of this example is:

$$y = f(x) = \sin(20x) + 5e^x - 20(x - 0.5)^2 + 2. \tag{5.1.1}$$

Suppose we want to find the maximum value of this function on the interval $[0, 1]$ without having access to the analytic form above and further suppose that every evaluation is very computationally expensive. The function is shown in Figure 5.1a. As described above, we first obtain some 'measurements', which in this case are function evaluations (assuming no noise). We use these to fit a regression term consisting of low-order polynomials. The order of the polynomial is bounded from above by the number of measurements and the number of input parameters. In Figure 5.1b the measurements are shown as well as the second order polynomial fitted to these measurements using least mean squares.

The next step is to construct a Gaussian process and train it on these measurements. A Gaussian process, once trained, will predict a both a mean and a variance for each input, unlike a regression model, which predicts a single value. The output of a Gaussian process, given a single input, is a distribution reflecting what we expect the quantity we are approximating to be, as well as how certain we are of this prediction. We train our Gaussian process by taking the residuals of our regression model (the 'measurements' minus the value predicted by the re-



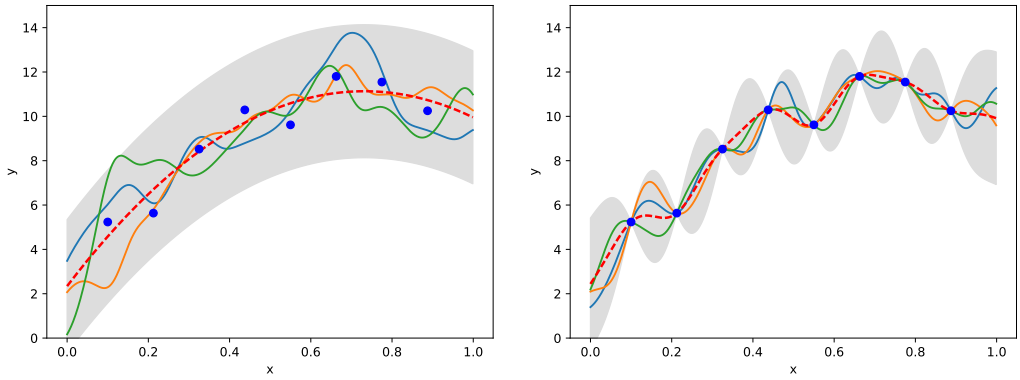
(a) The function from equation 5.1.1 which aim to emulate.

(b) The function from equation 5.1.1 in black, with 'measurements' shown by the blue dots and the regression model trained on these measurements shown by the red dotted line.

Figure 5.1: The function from equation 5.1.1 and the regression model trained on some measurements taken from this function.

gression model) and using these to parametrize a Gaussian process with mean 0. By then adding the predictions of the regression model and the Gaussian process, we get our final predicted mean. The standard deviation predicted by the Gaussian process tells us how sure we are of each prediction. In advance of a formal definition of a Gaussian process, which will be given in Section 5.2.2, think of a Gaussian process as follows: In the context of this example, a Gaussian process is a distribution over real-valued functions on $[0, 1]$, where each sample drawn from this distribution is a function that satisfies certain properties which depend on the covariance function we choose. In this case we choose a Gaussian (normal) distribution as covariance function, which means that the functions drawn from the Gaussian process are all smooth (infinitely differentiable). We work in a Bayesian framework so we specify a prior distribution for our Gaussian process and after adding in measurements we get a posterior distribution taking these into account. Figure 5.2a illustrates the prior distribution of a Gaussian process with a Gaussian correlation function with variance (which determines how far away from the regression model we expect values to be) of 1 and a correlation length (determines how strongly nearby function values correlate) of 0.05, as well as three sample functions. Sample functions are drawn by calculating the covariance matrix Σ for all x-coordinates and then calculating $\vec{y} = L\vec{z} + \mu(x)$ where $LL^T = \Sigma$ (L can be obtained by Cholesky decomposition), $\vec{z}$ is a vector of values pulled from independently identically distributed Gaussian distributions

and $\mu(x)$ is the mean predicted by the Gaussian process. Since this is the prior distribution and we specified a uniform prior, the Gaussian process predicts a standard deviation of 1 everywhere. After using the measurements to constrain the Gaussian process, we get a different distribution, as shown in Figure 5.2b. The predicted mean passes through each of the measurements and the predicted standard deviation is smaller for points closer to measurements since the nearby points constrain the possible range of values. The sample functions drawn from the Gaussian process pass through the measurements as well.



(a) The 'measurements' are shown in blue, the mean, in this case the regression model prediction, shown by the red dashed line. The shaded area is the 3σ interval around the mean. Three samples drawn from the Gaussian process are shown in blue, green and orange. Note that this is our prior distribution, so the measurement have not been taken into account by the Gaussian process yet.

(b) The measurements are shown in blue, the predicted mean is shown by the red dashed line. The shaded area is the 3σ interval around the mean. Three samples drawn from the Gaussian process are shown in blue, green and orange. Since this is the posterior distribution and we assumed there is no noise in our measurements, all samples pass through the measurements.

Figure 5.2: The prior and posterior distributions of an example Gaussian process added to the regression shown in Figure 5.1b, which aims to emulate the function from equation 5.1.1.

In the final step, we restrict parameter space by omitting all the points where we can be reasonably sure that the function value is below some threshold value T. By reasonably sure, we mean that $\mu(x) + 3\sigma(x) < T$, where $\mu(x)$ is the predicted mean at x and $\sigma(x)$ is the predicted standard deviation at x. Choosing 11 as threshold value, Figure 5.3 shows which parts of parameter space would be deemed 'interesting', further measurements would be restricted to these parts of parameter space. Note that both the original function and the predicted mean can be quite far below the threshold even in parts of parameter space that are deemed 'interesting'. This is because the predicted variance is high for these

points, so even though we think the mean is low in these areas, we are not certain that we should exclude them from consideration.

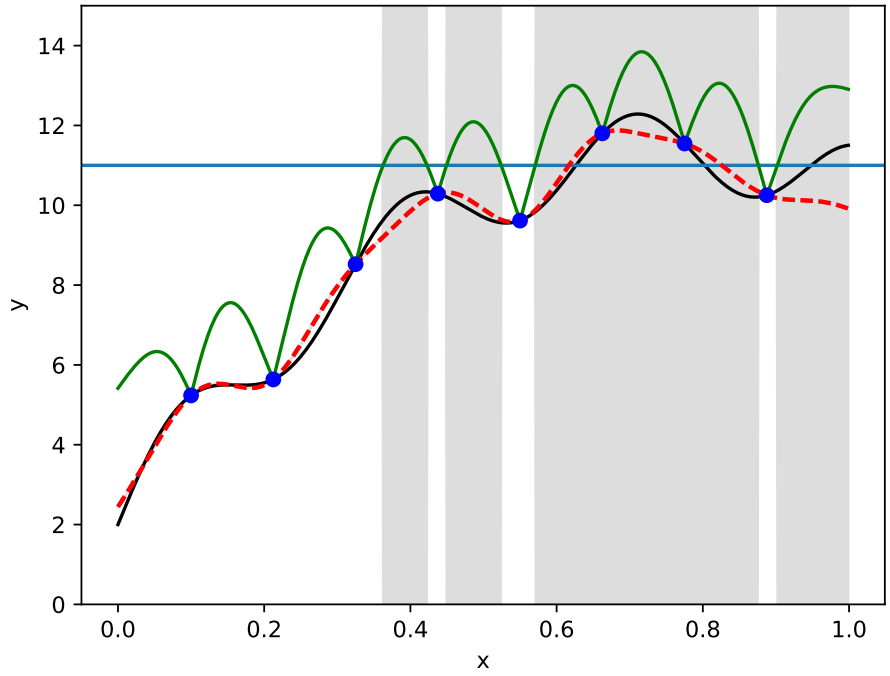


Figure 5.3: The original function, given by equation 5.1.1 is shown in black, the predicted mean is the red dashed line and the predicted mean plus 3σ is shown in green. Measurements are shown in blue and the shaded areas indicate the parts of parameter space where $\mu + 3\sigma \geq 11$.

The above illustrates how a first wave of simulations followed by constructing and using a wave-1 emulator looks in one dimension. After this, we can repeat the procedure on the reduced parameter space in successive waves to zoom in on the areas of parameter space where we expect the highest function values to be. When parameter space is high-dimensional, we need to make some adjustments to the procedure described above. Whereas in low dimensions it is possible to get a good initial set of datapoints with a regular grid, this is not possible in high-dimensional spaces. We will explain how we address this in section 5.3. As the number of inputs grows, the probability that some inputs have a very small effect on the output increases. To keep model complexity down, we will use only the inputs with the highest explanatory power to construct our regression model and Gaussian process, as explained in section 5.2.1.

5.2 Emulator Construction

We have a model $\vec{M}$, OpenSimRoot, which maps a vector of inputs $\vec{x}$ to a vector of outputs $\vec{M}(\vec{x})$. In this case, we want to maximise the shoot biomass so we will define $y(\vec{x})$ as the shoot biomass after 42 days for a vector of inputs $\vec{x}$. We pick 42 days because the OpenSimRoot maize root growth parameters have been measured up to 42 days. We construct an emulator of the form

$$f(\vec{x}) = \sum_i \beta_i g_i(\vec{x}) + u(\vec{x}) + w(\vec{x}). \tag{5.2.1}$$

Here β_i are constants determined by regression, g_i are basis functions (low-order polynomials), u is a Gaussian process over x and w is a nugget that is a white noise process uncorrelated to β_i.

5.2.1 Regression

The first term in our emulator, $\sum_i \beta_i g_i(\vec{x})$, is a regression model over low-order polynomials fitted using least squares. This is used to capture the global parameter dependence of our model. We expect polynomials to adequately approximate the large-scale structure of the OpenSimRoot response surface, in addition to having the following nice properties. Polynomials fitted using least squares are "non-local", which means that the predicted value at x can depend strongly on values far away from x. This property means we can build a global approximation for OpenSimRoot outputs but also that we should be cautious when making predictions far away from the training data. By spreading our initial measurements out over parameter space in such a way that no point is very far away from training data we aim to minimise extrapolation inaccuracies. Polynomials are easily interpretable, helping us interpret results. If a parameter explains a lot of the variation in the output variable, we should see it in the linear and quadratic terms for that parameter. Likewise, important interactions between parameters can be seen in the crossterms. We expect certain inputs to be more important for plant development than others and expect this to be reflected in the regression parameters.

Of course, there are other methods to approximate a multidimensional function

that could be used here, such as splines or radial basis functions. Splines and radial basis functions are both "local", which means they are less well suited for extrapolation to new parts of parameter space. We are also already taking into account local variations, namely through the second term of our emulator, the Gaussian process. In some cases, prior knowledge about the model being approximated can inform the choice for a particular method. In our case, the interpretability of polynomial regression, along with the absence of strong reasons for choosing a different method, make this our method of choice.

We need to specify a maximum degree for the polynomial regression we construct. If we choose a higher degree then we will also have more degrees of freedom, which should be lower than the amount of training data we have, ideally by a comfortable margin. As we will see in section 5.6, we will have 17 input parameters and 200 training data points. This means that we can at most fit a second degree polynomial, which has $1 + 17 + 17 + \frac{17 \cdot 16}{2} = 171$ degrees of freedom.

A second important consideration is which of the possible terms for a given set of inputs and degree to include. We decide the maximum degree we want to use in our regression polynomial and then use all the polynomial terms we can construct up to this degree. For 2 inputs x_1, x_2 and degree 2, we would have the terms: $1, x_1, x_2, x_1^2, x_1 x_2, x_2^2$. However, in general, we might not want to use all these terms in our regression. While using more terms can only improve the accuracy, this also makes the model more complicated in that it introduces more parameters to be fitted. If a model has a lot of parameters there is a risk of "overfitting", which means the model is able to reproduce the training data very accurately but does not provide accurate predictions on new inputs. To account for more than just the accuracy, but also the simplicity of the model, a commonly used metric is the Bayes information criterion (BIC) [178]. It is designed as follows:

$$BIC = -2\ln(\hat{\mathcal{L}}) + p \cdot \ln(k), \tag{5.2.2}$$

where p is the number of parameters included in the analysis, k is the number of data points and $\hat{\mathcal{L}}$ is the maximised likelihood function. By minimising this expression we find a balance between prediction accuracy and model complexity. This is because an additional parameter needs to increase the log of the maximised

likelihood function by at least $\frac{\ln(k)}{2}$. In order to take the maximum likelihood approach to regression, we assume that

$$y = h(\vec{x}) + \epsilon, \tag{5.2.3}$$

where $h(\vec{x}) = \sum_i \beta_i g_i(\vec{x})$ is a sum of polynomials in $\vec{x}$ and ϵ is a random variable which captures the variation not explained by $h(\vec{x})$, which we model as random noise. Different choice of noise are possible for ϵ but a Gaussian distribution with a mean of zero is a natural choice. Using this, the likelihood function for a single prediction $h(\vec{x})$, is

$$L(h(\vec{x})) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(h(\vec{x})-y)^2}{2\sigma^2}}, \tag{5.2.4}$$

where $h(\vec{x})$ is our prediction, y is the 'true' or measured value and σ is the standard deviation. To get the likelihood function for the entire data set, we assume that the noise terms ϵ for different inputs are independent and identically distributed, which means that the likelihood given multiple data points is the product of the individual likelihood functions. So we multiply the individual likelihood functions:

$$L(h) = \prod_i \left[\frac{1}{\sqrt{2\pi\sigma_i^2}} e^{-\frac{(h(\vec{x}_i)-y_i)^2}{2\sigma_i^2}} \right] = \prod_i \left[\frac{1}{\sqrt{2\pi\sigma_i^2}} \right] e^{-\sum_i \frac{(h(\vec{x}_i)-y_i)^2}{2\sigma_i^2}}. \tag{5.2.5}$$

Note that adjusting $h(\vec{x})$ to maximise this likelihood function is the same as using least squares, since in both cases $\sum_i (h(\vec{x}_i) - y_i)^2$ is minimised. Since we use BIC to compare different regressions on the same dataset, we can ignore the constant terms when we take the log of the maximised likelihood, so we get

$$BIC = \sum_i \frac{(h(\vec{x}_i) - y_i)^2}{\sigma_i^2} + p \cdot \ln(k). \tag{5.2.6}$$

In cases where we have, say, 10 input parameters and want to include terms up to degree 2, we would have $1 + 10 + 10 + 45 = 66$ (1 constant term c, 10 linear terms x_i, 10 quadratic terms x_i^2, 45 quadratic terms $x_i x_j$ with $i \neq j$) different terms, including the constant term. Assuming the constant term is always included, we would have to calculate the BIC for $2^{65} \approx 3.7 \cdot 10^{19}$ different combinations of included input parameters to find the optimal combination. Even if we could

evaluate 10^{12} combinations of inputs every second, this would still take more than a year. So instead of going through every combination, we will either start without any terms and add the one that improves the BIC the most, one at a time, or start with all terms, and remove terms one by one, until we cannot improve our metric anymore by removing or adding a single term. In this way, we would have to test at most $\sum_{i=0}^{65} 65 - i = \frac{65 \cdot 66}{2} = 2145$ cases, which is much more computationally tractable. Note that we might get a different set of terms that maximises the BIC depending on whether we add terms one by one or substract them one by one. In any case, human judgment factors in the final choice of active inputs.

5.2.2 Gaussian Process

The second term in our emulator, $u_i(\vec{x}_{A_i})$, is a Gaussian process. We will first give some definitions and then describe the construction of the Gaussian process from the data. Roughly, a Gaussian process is a way to choose, given some test data points, a distribution over all functions that is consistent with the test data points and provide a normal distribution for the possible outputs at every input point. It's the infinite-dimensional generalisation of multivariate Gaussian distributions. Let us first recall some definitions from probability theory.

Definition 5.1. Let X be a set and $\mathcal{P}(X)$ its power set. A subset $S \subset \mathcal{P}(X)$ is a σ-algebra if

1. $X \in S$.
2. If $A \in S$ then its complement $X \setminus A \in S$. In other words, S is closed under complementation.
3. If $A_1, A_2, ... \in S$ then $A = \cup_{i=1}^{\infty} A_i \in S$. In other words, S is closed under countable unions.

Definition 5.2. Let X be a set and S a σ-algebra over X. A function μ from S to the extended real number line ($\mathbb{R} \cup \{-\infty, +\infty\}$) is a measure if it satisfies:

1. Non-negativity: $\forall x \in S, \mu(x) \geq 0$
2. Null empty set: $\mu(\varnothing) = 0$
3. Countable additivity: For all countable collections $\{x_n\}_{n=1}^{\infty}$ of pairwise disjoint sets in S it is true that $\mu(\cup_{n=1}^{\infty} x_n) = \sum_{n=1}^{\infty} \mu(x_n)$.

A measurable set is a tuple (X, S) for which there exists a measure.

Definition 5.3. A probability space is a triple $(\Omega, \mathcal{F}, P)$ where

- the sample space Ω is a non-empty set.
- $\mathcal{F}$ is a σ-algebra on Ω.
- $P : \mathcal{F} \to [0, 1]$ is a measure on $\mathcal{F}$ for which it is true that $P(\Omega) = 1$.

Definition 5.4. Given a probability space $(\Omega, \mathcal{F}, P)$ and a measurable space (X, S), a stochastic process is a collection of X-valued random variables on $(\Omega, \mathcal{F}, P)$, indexed by some set T. We write this as:

$$\{X(t) : t \in T\}. \tag{5.2.7}$$

Definition 5.5. A random vector $X = \begin{pmatrix} X_1 & X_2 & \dots & X_n \end{pmatrix}^T$ has a multivariate normal distribution if every linear combination $Y = a_1X_1 + ... + a_nX_n$ is normally distributed.

In other words, a multivariate normal distribution is a vector of jointly normally distributed random variables. The probability density of a multivariate normal distribution $X \cong \mathcal{N}(\mu, \Sigma)$ is equal to

$$P(x; \mu, \Sigma) = \frac{e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}}{\sqrt{(2\pi)^n|\Sigma|}}. \tag{5.2.8}$$

Here x is an n-dimensional vector, Σ is the covariance matrix and $|\Sigma|$ is the determinant of Σ.

Definition 5.6. A stochastic process $\{X(t) : t \in T\}$ is Gaussian if and only if for every finite set of indices, $t_1, ..., t_n$ in the index set T, $X_{t_1,...,t_n} = (X(t_1), ..., X(t_n))$ is a multivariate Gaussian random variable.

A Gaussian process is the generalisation of a Gaussian random vector to an uncountably infinite index set. Another way to view it is as a distribution over functions, where each sample from the distribution is a function over the index set (in our case this is $\mathbb{R}^n$) that satisfies some conditions that derive from the covariance we assign to pairs of points.

Let X be some set that contains the possible values of the input parameters to our model (e.g. $X \subset \mathbb{R}^n$ is a set of times/locations/growth rates/reaction coefficients). We have some system $M : X \to \mathbb{R}$ that we want to predict for any $\vec{x} \in X$ given a set of observations $\{(\vec{x}_1, y_1), ..., (\vec{x}_k, y_k)\}$. Now we want to predict $M(\vec{x}_*)$ for some $\vec{x}_* \in X$. Using the method of Gaussian processes, we assume that our set of observations $\vec{y} = \begin{pmatrix} y_1 & y_2 & \dots & y_k \end{pmatrix}^T$ can be represented as a sample from a multivariate Gaussian distribution, which means that

$$\vec{y} \sim \mathcal{N}(\vec{\mu}, \Sigma), \tag{5.2.9}$$

where $\vec{\mu}$ is the vector containing all the means μ_i for each observation and Σ is a symmetric positive definite matrix which has the variances σ_i on the diagonal and the covariances $\sigma_{ij} = \sigma(\vec{x}_i, \vec{x}_j)$ as off-diagonal elements. We need to choose a covariance function, which we expect to help us model the system well. Since the covariance function determines how values at nearby points are related, the choice of covariance function puts constraints on the behaviour of the Gaussian process. The 'default' kernel of choice is the squared exponential (or "exponentiated quadratic" kernel), which has covariance function

$$\sigma(\vec{x}, \vec{z}) = \sigma_m^2 e^{-\sum_{i=1}^{n} \frac{(x_i - z_i)^2}{2l_i^2}}, \tag{5.2.10}$$

where σ_m is the maximum allowable covariance and l_i is the correlation length for the i-th parameter, which will determine how fast the function can vary. This covariance function has several properties which make it suitable for our purposes: It is universal [129], which means that it can be used to approximate any continuous target function uniformly on any compact subset of the input space. Using this kernel, the samples drawn from the Gaussian process are smooth (infinitely differentiable), which we think is a reasonable assumption for OpenSimRoot outputs to be. In some cases, a different kernel would be more suitable, such as if the model to be approximated has a periodic behaviour. This is not the case for us so the squared exponential kernel suits our purposes. To get a prediction y_* for some given x_*, we remember that $\begin{pmatrix} \vec{y} \\ y_* \end{pmatrix}$ is a sample from a multivariate

Gaussian random variable (by definition), so

$$\begin{pmatrix} \vec{y} \\ y_* \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \vec{\mu} \\ \mu_* \end{pmatrix}, \begin{pmatrix} \Sigma & \Sigma_* \\ \Sigma_*^T & \Sigma_{**} \end{pmatrix} \right), \tag{5.2.11}$$

where $\Sigma_* = \begin{pmatrix} \sigma(\vec{x}_*, \vec{x}_1) & \cdots & \sigma(\vec{x}_*, \vec{x}_n) \end{pmatrix}^T$ and $\Sigma_{**} = \sigma(\vec{x}_*, \vec{x}_*)$. For the more general case, we can write down the conditional density:

$$p(\vec{y}_1|\vec{y}_2) = \frac{p(\vec{y}_1, \vec{y}_2; \mu, \Sigma)}{\int_{\vec{y}_1} p(\vec{y}_1, \vec{y}_2; \mu, \Sigma) d\vec{y}_1} \tag{5.2.12}$$

$$= \frac{1}{\int_{\vec{y}_1} p(\vec{y}_1, \vec{y}_2; \mu, \Sigma) d\vec{y}_1 \sqrt{(2\pi)^n |\Sigma|}} \exp\left(-\frac{1}{2} (\vec{y} - \vec{\mu})^T \Sigma^{-1} (\vec{y} - \vec{\mu}) \right) \tag{5.2.13}$$

$$= \frac{1}{Z_1} \exp\Bigg(-\frac{1}{2} \big[(\vec{y}_1 - \vec{\mu}_1)^T V_{11} (\vec{y}_1 - \vec{\mu}_1) + (\vec{y}_1 - \vec{\mu}_1)^T V_{12} (\vec{y}_2 - \vec{\mu}_2) \tag{5.2.14}$$

$$+ (\vec{y}_2 - \vec{\mu}_2)^T V_{21} (\vec{y}_1 - \vec{\mu}_1) + (\vec{y}_2 - \vec{\mu}_2)^T V_{22} (\vec{y}_2 - \vec{\mu}_2) \big] \Bigg) \tag{5.2.15}$$

$$= \frac{1}{Z_2} \exp\left(-\frac{1}{2} \left[\vec{y}_1^T V_{11} \vec{y}_1 - 2\vec{y}_1^T V_{11} \vec{\mu}_1 + 2\vec{y}_1^T V_{12} (\vec{y}_2 - \vec{\mu}_2) \right] \right) \tag{5.2.16}$$

$$= \frac{1}{Z_3} \exp\left(-\frac{1}{2} \left[\left(\vec{y}_1 - \vec{\mu}'\right)^T V_{11} \left(\vec{y}_1 - \vec{\mu}'\right) \right] \right). \tag{5.2.17}$$

Here we used that $V_{12} = V_{21}^T$ (which follows from the fact that Σ^{-1}, being the inverse of a covariance matrix, is symmetric), Z_1, Z_2 and Z_3 are constants that do not depend on $\vec{y}_1$ and $\vec{\mu}' = \vec{\mu}_1 - V_{11}^{-1} V_{12} (\vec{y}_2 - \vec{\mu}_2)$ and $\Sigma^{-1} = \begin{pmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix}$. From this it follows that

$$\vec{y}_1|\vec{y}_2 \sim \mathcal{N}(\vec{\mu}_1 - V_{11}^{-1} V_{12} (\vec{y}_2 - \vec{\mu}_2), V_{11}^{-1}). \tag{5.2.18}$$

Now we note that

$$\begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}^{-1} \tag{5.2.19}$$

$$= \begin{pmatrix} (\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})^{-1} & -(\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})^{-1}\Sigma_{12}\Sigma_{22}^{-1} \\ -\Sigma_{22}^{-1}\Sigma_{21}(\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})^{-1} & \Sigma_{22}^{-1} + \Sigma_{22}^{-1}\Sigma_{21}(\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})^{-1}\Sigma_{12}\Sigma_{22}^{-1} \end{pmatrix} \tag{5.2.20}$$

$$= \begin{pmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix}. \tag{5.2.21}$$

Inserting this into equation 5.2.18, we get

$$\vec{y}_1|\vec{y}_2 \sim \mathcal{N}\Big(\vec{\mu}_1 + (\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})(\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})^{-1}\Sigma_{12}\Sigma_{22}^{-1}(\vec{y}_2 - \vec{\mu}_2), \tag{5.2.22}$$

$$(\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})\Big) \tag{5.2.23}$$

$$= \mathcal{N}(\vec{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\vec{y}_2 - \vec{\mu}_2), (\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})). \tag{5.2.24}$$

From this it follows that

$$y_*|\vec{y} \sim \mathcal{N}(\mu_* + \Sigma_*^T\Sigma^{-1}(\vec{y} - \vec{\mu}), \Sigma_{**} - \Sigma_*^T\Sigma^{-1}\Sigma_*). \tag{5.2.25}$$

The above expression gives us the mean and standard deviation of our Gaussian process for any $\vec{x}$. We can simplify the above equations by taking a distribution with mean 0 as prior, so $\mu(\vec{x}) = 0$ for all $\vec{x} \in X$. This simplifies our equation to

$$\begin{pmatrix} \vec{y} \\ y_* \end{pmatrix} \sim \mathcal{N}\left(\vec{0}, \begin{pmatrix} \Sigma & \Sigma_* \\ \Sigma_*^T & \Sigma_{**} \end{pmatrix}\right), \tag{5.2.26}$$

$$y_*|\vec{y} \sim \mathcal{N}(\Sigma_*^T\Sigma^{-1}\vec{y}, \Sigma_{**} - \Sigma_*^T\Sigma^{-1}\Sigma_*). \tag{5.2.27}$$

We use this expression by first 'renormalising' our vector of observations $\vec{y}$: Define $y_i' = y_i - h(\vec{x}_i)\forall i$, where $h(\vec{x}_i)$ is the prediction of the regression polynomial described in the previous section. To simplify notation, we will denote the renormalised observations by $\vec{y}$ from now on. For a given input $\vec{x}$, a Gaussian process predicts the following mean μ and variance σ

$$\mu(\vec{x}) = \Sigma_*^T\Sigma^{-1}\vec{y}, \tag{5.2.28}$$

$$\sigma^2(\vec{x}) = \Sigma_{**} - \Sigma_*^T\Sigma^{-1}\Sigma_*. \tag{5.2.29}$$

Observation noise

The final term in our emulator, $w_i(\vec{x})$ is a noise term that accounts for the fact that our observations may not be perfect (in our case, the system we are trying to model, OpenSimRoot has a stochastic component) and we only consider some inputs as "active inputs", the others will contribute to this noise term. Usually

this is done by modelling observations as

$$y = f(x) + \epsilon, \tag{5.2.30}$$

where ϵ is additive independent identically distributed Gaussian noise with variance θ^2. In our case, we add a noise term

$$\theta(\vec{x}_i, \vec{x}_j) = \alpha^2 \delta_{\vec{x}_i, \vec{x}_j}, \tag{5.2.31}$$

to our covariance function, where $\delta_{\vec{x}_i, \vec{x}_j}$ is the kronecker delta function. In the discretised case, we add $\alpha^2 I$ to our covariance matrix. In other words, our covariance is given by

$$cov(\vec{x}_i, \vec{x}_j) = \sigma(\vec{x}_i, \vec{x}_j) + \alpha^2 \delta_{\vec{x}_i, \vec{x}_j}, \tag{5.2.32}$$

or, in the discretised case

$$\Sigma' = \Sigma + \alpha^2 I. \tag{5.2.33}$$

Then the joint distribution of the observed target values and the test values is:

$$\begin{pmatrix} \vec{y} \\ y_* \end{pmatrix} \sim \mathcal{N}\left(\vec{0}, \begin{pmatrix} \Sigma + \alpha^2 I & \Sigma_* \\ \Sigma_*^T & \Sigma_{**} \end{pmatrix}\right), \tag{5.2.34}$$

$$y_* | \vec{y} \sim \mathcal{N}(\Sigma_*^T (\Sigma + \alpha^2 I)^{-1} \vec{y}, \Sigma_{**} - \Sigma_*^T (\Sigma + \alpha^2 I)^{-1} \Sigma_*). \tag{5.2.35}$$

With this, equations 5.2.28 and 5.2.29 become

$$\mu(\vec{x}) = \Sigma_*^T (\Sigma + \alpha^2 I)^{-1} \vec{y}, \tag{5.2.36}$$

$$\sigma^2(\vec{x}) = \Sigma_{**} - \Sigma_*^T (\Sigma + \alpha^2 I)^{-1} \Sigma_*. \tag{5.2.37}$$

We have now completed constructing the emulator as described in equation 5.2.1 with the polynomial regression accounting for the first term and the Gaussian process accounting for the final two. Because the emulator not only predicts a mean value but also provides a corresponding variance, we have an estimate of how certain each prediction is. This is an important feature that we will use to be very careful in how we evaluate the predictions supplied by the emulator.

5.2.3 Choosing hyperparameters

Parameters which are used to control the learning process in machine learning are called hyperparameters. In our case, the kernel functions determines how the Gaussian process assimilates training data, so the kernel parameters are the hyperparameters. When constructing a Gaussian process, choosing the hyperparameters is very important. If the correlation length is too low, the Gaussian process will predict zero everywhere except very close to any training points and if the correlation length is too high, results will influence regions of parameter space that are not alike at all. Likewise, if the variance is too high, the mean does not have to vary much for all measurements to fall within acceptable bounds, which means the Gaussian process will not be responsive to the training data. If the variance is too low, the mean will move to match every training point very closely and overfit itself on the data, missing the larger picture. For a full Bayesian approach here we would define prior distributions for our hyperparameters and then use Bayes' theorem to calculate the posterior distribution given the data we have. Since we have multiple correlation lengths, this would likely not be analytically tractable so we would have to use numerical methods to approximate the posterior distribution. However, since the Gaussian process only explains part of the variation, we assume that any uncertainty on the exact value of the hyperparameters will not be critical. So instead of doing a full Bayesian analysis, we instead use maximum likelihood to determine the values of the hyperparameters. Note that this is equivalent to doing a Bayesian analysis with a uniform prior and choosing the maximum a posteriori probability estimate. We are trying to find the hyperparameters θ' for which

$$\theta' = \arg\max_{\theta} L(\theta|Y). \tag{5.2.38}$$

From [44], we have the following expression for the log likelihood of a Gaussian process with noise:

$$\mathcal{L}(\theta|\vec{y}) = -\frac{1}{2}\left[(\vec{y}-\vec{\mu})^T(\Sigma+\alpha^2 I)^{-1}(\vec{y}-\vec{\mu}) + \log|\Sigma+\alpha^2 I| + n\log(2\pi)\right]. \tag{5.2.39}$$

Here $\vec{\mu}$ is the mean of the Gaussian process, which in our case is the value predicted by the linear regression. By maximising this over θ we find the hyperparameters with the highest likelihood. In our case, we do this numerically using the SciPy Python package.

5.3 Exploring Parameter Space

Now that we know how to construct our emulator, we can describe the procedure. We start with a parameter space $P \subset \mathbb{R}^n$ and some model $M : P \to \mathbb{R}$. We want to find the subset $Q \subset P$ for which $M(q)$ for $q \in Q$ is minimised or maximised with respect to some metric such as closeness of fit to observed data or, in our case, maximising a specific output. We start with an initial set of simulations $\{M(\vec{x}_1), ..., M(\vec{x}_k)\}$ that we will use to construct our wave-1 emulator.

The choice of initial inputs $I_0 = \{\vec{x}_1, ..., \vec{x}_k\} \subset P$ is not entirely trivial. They should provide good coverage of P, that is,

$$\max_{\vec{x} \in P} \{ \min_{\vec{x}_i \in I_0} \{d(\vec{x}, \vec{x}_i)\}\} \tag{5.3.1}$$

should be 'small enough'. Here, $d(\vec{x}, \vec{y})$ is the distance between $\vec{x}$ and $\vec{y}$. Choosing an evenly spaced grid is a simple way to achieve this in low dimensions but in higher dimensions this requires a very large number of samples. Even with just 2 different values per input parameter, we would need $2^{10} = 1024$ samples if we have 10 input parameters. We could just randomly take as many sample as our computing resources allow, but this might leave large areas of parameter space unsampled. Instead of this, we sample parameter space according to a Latin hypercube design. In a Latin hypercube design with k samples, each dimension of parameter space is divided into k segments of equal length. The samples are then chosen in such a way that when they are projected unto any single axis, each of these k segments will have exactly one sample in them, but their location within each segment is chosen randomly. To ensure good coverage, we create some large number of Latin hypercubes and choose the hypercube H such that the quantity

$$\min_{\vec{x}_i, \vec{x_j} | i \neq j} \{d(\vec{x}_i, \vec{x}_j) | \vec{x}_i, \vec{x}_j \in H\} \tag{5.3.2}$$

is maximised. Note that, prior to creating these hypercubes, we rescale parameter space to $[-1, 1]^p$, where p is the dimension of parameter space. For some parameters, a linear scaling is most appropriate, while for others a logarithmic scaling is more appropriate. Some parameters might be discrete, but enumerated, e.g. they take values in some subset $S \subset \mathbb{Z}$. In this case we would restrict the possible values parameters can take in that dimension to a discrete number of values. If a parameter can take discrete values that do not have a linear relation (for example, a "plant species" parameter might have the possible values {barley, bean, maize}), we can do either of two things, depending on the output dependence on this parameter. If this parameter has a 'small' influence on the output, we can add a term that depends on this parameter to our emulator. If the difference is 'large', we have to construct different emulators for different values of this parameter.

Once we have constructed our emulator, we can use it to make predictions of our model response. If parameter space is low-dimensional, we can evaluate our emulator on a fine grid spanning all of parameter space. For high-dimensional parameter spaces this is usually not feasible, so we sample a large number of points ($\pm 10^9$) which can be chosen in a variety of ways (e.g. randomly, Latin hybercubes, etc.). For each point $\vec{x} \in P$, our emulator predicts a mean $\mu(\vec{x})$ and a variance $\sigma(\vec{x})$. This means we do not just get a prediction for our variable of interest, but we can also quantify how certain we are about these predictions.

According to our emulator, the 'true' value at $\vec{x}$, $M(\vec{x})$, is a random variable X following a distribution with mean $\mu(\vec{x})$ and variance $\sigma(\vec{x})$. Since we do not know the shape of this distribution, we have to make an assumption here. A common choice is to assume that X follows a normal distribution, strictly following the Gaussian process definition, but we make the weaker and more robust assumption that X is distributed unimodally (the distribution has only one peak) [196, 197]. The Pukelsheim 3σ rule [163] tells us that, in this case,

$$P\Big(|X - \mu(\vec{x})| \geq 3\sigma(\vec{x})\Big) \leq \frac{4}{81} < 0.05. \tag{5.3.3}$$

Based on this we define the 'interesting' region of parameter space as

$$P_I := \{\vec{x} \in P \mid (\mu(\vec{x}) - 3\sigma(\vec{x}), \mu(\vec{x}) + 3\sigma(\vec{x})) \cap R_I \neq \emptyset\}. \tag{5.3.4}$$

Here $R_I \subset \mathbb{R}$ is the set of outputs that we think are 'good enough'. The points in the interesting set are those for which the predicted output is close than 3σ to R_I. In our case, we want to find the set of parameters that result in a high yield (shoot dry weight), so R_I will be of the form $[R_{min}, \infty)$ for some shoot dry weight value R_{min}. So P_I is defined as

$$P_I := \{\vec{x} \in P \mid \mu(\vec{x}) + 3\sigma(\vec{x}) \geq R_{min}\}. \tag{5.3.5}$$

5.4 Emulator diagnostics

Diagnostics are used to make sure that our emulator is behaving the way we expect. One way is to cross-validate by dividing the data in two parts; a training set and a validation set. Then we construct an emulator with the training set and check how well it can predict the validation set. By choosing different subsets of the data as training and validation sets we can do multiple cross-validations of our emulator construction. If the validation set is a single data point, this is called leave-one-out cross-validation. This way our emulator is very similar to the emulator using all the data and we can test if we are able to accurately predict each data point[1].

After training the emulator on all data except one point $\vec{x}_i$, we calculate the normalised variance

$$v_n(\vec{x}_i) = \frac{|y_i - \mu(\vec{x}_i)|}{\sigma(\vec{x}_i)}. \tag{5.4.1}$$

By calculating the fractions of test points for which v_n is larger than $1, 2$ or 3, denoted by $v_n^{>1}$, $v_n^{>2}$ and $v_n^{>3}$, we can evaluate our predictions. For values distributed around μ according to a Gaussian distribution, we expect $v_n^{>1} \approx 0.3173$,

[1] Recall equations 5.2.28 and 5.2.29. We need to invert the covariance matrix for the Gaussian process to make predictions. Since we already have the inverse of the full covariance matrix, we can use the Woodbury matrix identity $(A + UCV)^{-1} = A^{-1} - A^{-1}U(C + VA^{-1}U)^{-1}VA^{-1}$ to calculate the necessary inverses in $\mathcal{O}(n^2)$ operations instead of $\mathcal{O}(n^3)$. See Appendix C for more details.

$v_n^{>2} \approx 0.0455$ and $v_n^{>3} \approx 0.0027$. If we make the weaker assumption that the distribution is unimodal, the Vysochanskij–Petunin inequality [163] tells us that $v_n^{>2} \leq \frac{1}{9}$ and $v_n^{>3} \leq \frac{4}{81}$.

One downside to this method is that points in our training set are far apart from each other, by design. This provides good coverage of parameter space but also means that our reduced emulators will have difficulty making these predictions.

We can also check our emulator for accuracy by having a training set used to construct it and then check predictions against a separate validation set. The downside to this is that we need to use computing resources for additional simulations that are not optimised for use in constructing an emulator. This is because this additional validation set would not part of the same Latin hypercube design as our training set. Still, this gives us more data to train our emulator on, after verifying that it is working as intended.

5.5 Integrating 20-day simulations into the emulator

Since the time and memory needed for each timestep in OpenSimRoot increase with root system size, a 20-day simulation takes considerable less time than a 42-day simulation[2]. We can use this to our advantage by running a 'wave 1.5'; a wave of 20-day simulation meant to refine our emulator before doing an expensive second wave of 42-day simulations. With this as our aim, we want this in-between wave to help us exclude as much parameter space as possible. This is why the locations of these simulations will be those $\vec{x}$ where $\mu(\vec{x}) + 3\sigma(\vec{x})$ is close to our cutoff threshold and where $\sigma(\vec{x})$ is high. In other words, we want to do them in locations where we are not very confident in our predictions and where we are most unsure if we want to include or exclude it. To make this more quantitative, to a point $\vec{x}$ in parameter space we assign the weight

$$\omega(\vec{x}) = e^{-\frac{(T-(\mu(\vec{x})+3\sigma(\vec{x})))^2}{\sigma^2(\vec{x})}}. \tag{5.5.1}$$

[2]After completing all OpenSimRoot simulations for this chapter we were able to quantify this. 42-day simulations took about 16.3 hours on average, while 20-day simulations took about 1.37 hours on average, a difference of more than a factor 10!

With T our cutoff value. This weight ensures that we care the most about the variances of the points that are near our cutoff, in other words, the points that we are least sure about. Now we define $\sigma_{\vec{z}}(\vec{x})$ as the variance a Gaussian process would predict, if we added a measurement at $\vec{z}$ to our training set. This is possible because as we see from equation 5.2.37, $\sigma(\vec{x})$ depends only on the covariances between points, not the actual results at those points. We generate and fix a large number of reference points X that cover parameter space. Then we add inputs $\vec{z}$ to the set of wave 1.5 input locations P one by one by finding the $\vec{z}$ which minimises

$$\nu(\vec{z}) := \sum_{\vec{x} \in X} \sigma^2_{P,\vec{z}}(\vec{x})\omega(\vec{x}). \tag{5.5.2}$$

Here $\sigma^2_{P,\vec{z}}(\vec{x})$ is the variance predicted by a Gaussian process that includes P and $\vec{z}$ as training data (as well as the data we already had). Note that this is not necessarily the set of points that minimises our weighted variance function the most, since we are adding points one by one, but it would be too computationally expensive to minimise over multiple points at the same time.

We will use the results from these 20-day simulations by fitting a polynomial regression, refered to as the forward regression, to our first wave of 42-day simulations to train this regression model. We can do this because OpenSimRoot provides outputs for relevant state variables at specified intervals, in our case daily. We have chosen 25 state variables expected to be good early indicators of plant performance, such as total root length, total nitrogen, phosphorus and potassium uptake, shoot size, rooting depth, see Table B.7 for the complete list. The relationship between these variables at day 20 and the shoot dry weight at day 42 is shown in figure 5.4.

For a location $\vec{x}$ we estimate the shoot dry weight at day 42, $y_{42}(\vec{x})$ as

$$y_{42}(\vec{x}) = \kappa_0 + \sum_{i=1}^{k} \kappa_i y^i_{20}(\vec{x}) + \sum_{i \geq j}^{k} \kappa_{ij} y^i_{20}(\vec{x}) y^j_{20}(\vec{x}), \tag{5.5.3}$$

where κ_i, κ_{ij} are constants that we fitted to our wave 1 simulations and y^i_{20} the selected state variables at 20 days. This regression has an adjusted coefficient of

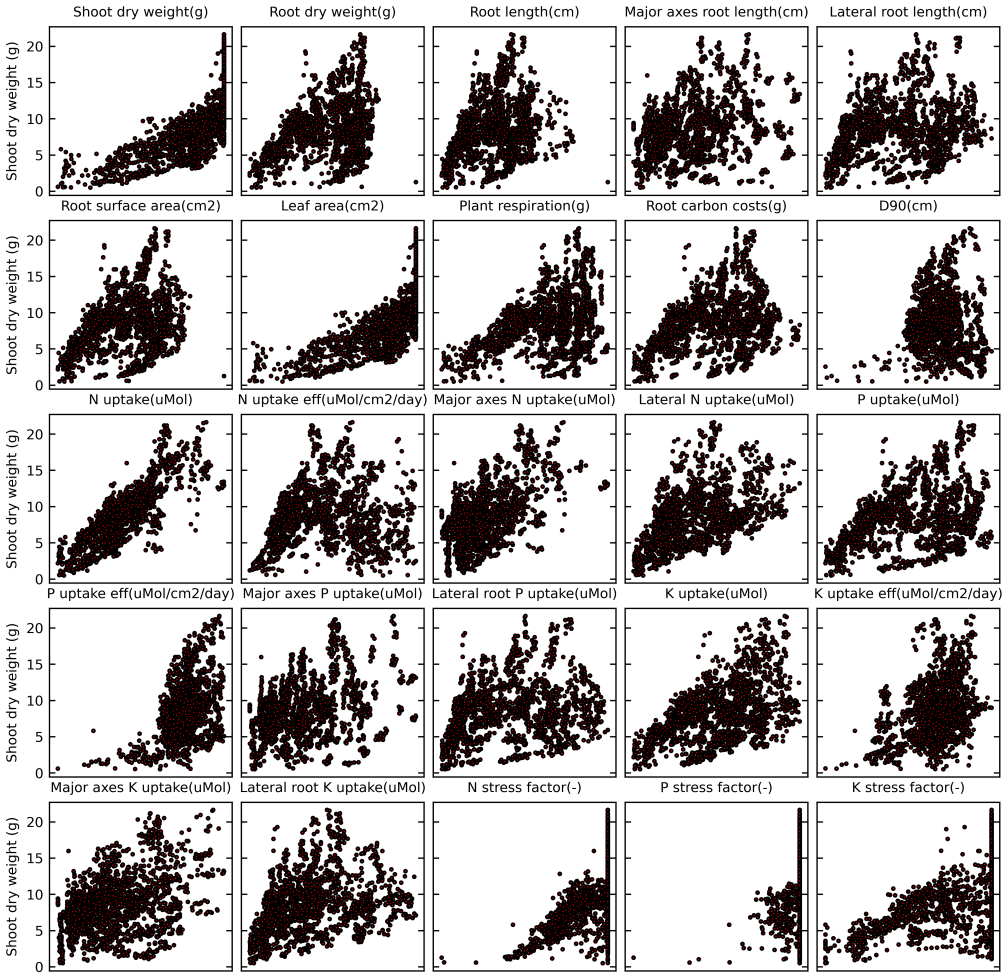


Figure 5.4: Relationship between maize shoot dry weight after 42 days and selected outputs on day 20 for the wave 1 simulations. Y-axes are shared between panels, the x-axes are not and are unlabeled to improve readability.

determination (r-squared) of 0.96575, but this predicted value is of course still less accurate than running a full 42-day simulation. Figure 5.5 shows the relationship between the 42-day shoot dry weight predicted from the outputs at day 20 and the simulated 42-day shoot dry weight. We want to integrate this prediction into our Gaussian process, taking this difference in accuracy into account. Denoting the residual variation of the above regression model by η^2, we do this as follows.

Since the value of the output variable predicted from a 20-day simulation is strongly correlated with the output we would get from a full 42-day simulation, we write

$$y_{42}(\vec{x}) = y(\vec{x}) + \epsilon(\vec{x}), \tag{5.5.4}$$

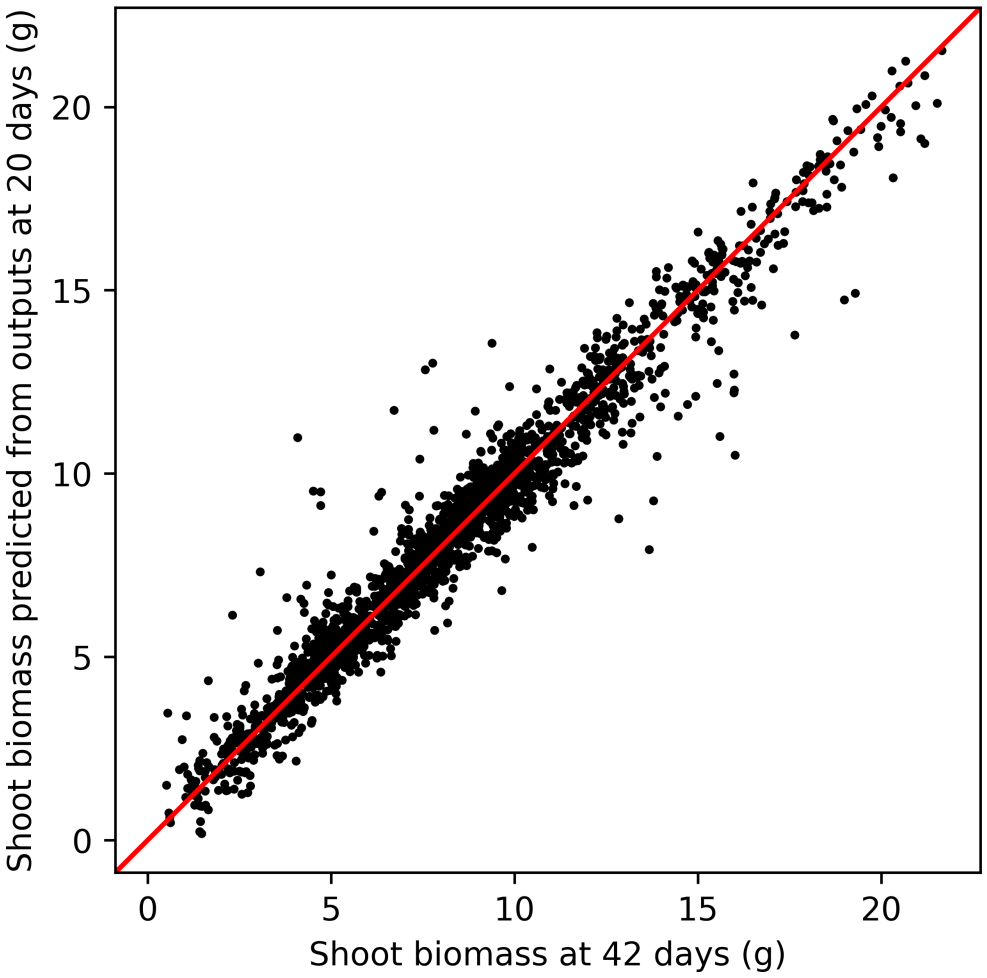


Figure 5.5: Relationship between maize shoot dry weight at 42 days as predicted from the outputs at day 20 and simulated maize shoot dry weight at 42 days.

where $y(\vec{x})$ is the 'real' value, which we would obtain by simulating the full 42 days and ϵ is a noise term that is independent of $\vec{x}$ and $y(\vec{x})$ such that

$$var(\epsilon(\vec{x})) = \eta^2, \tag{5.5.5}$$

$$cov(\epsilon(\vec{x}_1), \epsilon(\vec{x}_2)) = 0, \quad x1 \neq x2, \tag{5.5.6}$$

$$cov(\epsilon(\vec{x}_1), y(\vec{x}_2)) = 0, \tag{5.5.7}$$

where η^2 is the unexplained variance of the regression model linking 20-day simulations to outputs at 42 days, given by equation 5.5.3. Recall that we use the following covariance function

$$cov(y(\vec{x}_1), y(\vec{x}_2)) = \sigma^2 e^{-\sum_{i=1}^{n} \frac{(x_1^i - x_2^i)^2}{2l_i^2}} + \alpha^2 \delta_{\vec{x}_1, \vec{x}_2}. \tag{5.5.8}$$

With this we can calculate the covariances associated with 20-day simulations:

$$cov(\vec{y}(x_1), \vec{y}_{42}(x_2)) = cov(\vec{y}(x_1), \vec{y}(x_2) + \epsilon(\vec{x}_2)) \tag{5.5.9}$$

$$= \sigma^2 e^{-\sum_{i=1}^{n} \frac{(x_1^i - x_2^i)^2}{2l_i^2}} + \alpha^2 \delta_{\vec{x}_1, \vec{x}_2}. \tag{5.5.10}$$

$$cov(\vec{y}_{42}(x_1), \vec{y}_{42}(x_2)) = cov(\vec{y}(x_1) + \epsilon(x_1), \vec{y}(x_2) + \epsilon(x_2)) \tag{5.5.11}$$

$$= \sigma^2 e^{-\sum_{i=1}^{n} \frac{(x_1^i - x_2^i)^2}{2l_i^2}} + \alpha^2 \delta_{\vec{x}_1, \vec{x}_2} + \eta^2 \delta_{\vec{x}_1, \vec{x}_2}. \tag{5.5.12}$$

Now we use this to construct the wave 1.5 emulator. We define the vector of wave 1.5 inputs $\vec{y}_{1.5}$ as

$$\vec{y}_{1.5} = \begin{pmatrix} \vec{y} \\ \vec{y}_{42} \end{pmatrix}. \tag{5.5.13}$$

Using the expressions above we can construct the wave 1.5 covariance matrix $\Sigma_{1.5}$ and construct an emulator as before.

5.6 Optimising Shoot Dry Weight

We apply the methods described above to the problem of optimising a maize root phenotype for shoot biomass while subjected to nitrogen, phosphorus and potassium stress. From the literature we already know the optimal values for some phenes for plants subjected to either nitrogen or phosphorus stress. For example, in low nitrogen conditions, low lateral branching density is optimal, while high lateral branching density is optimal under low phosphorus [154]. In general, since phosphorus is immobile and concentrated in the topsoil, phenotypes with high root density in the topsoil perform better than those with low root density in the topsoil and competition between roots is not significant. Nitrate, on the other hand, is highly mobile so competition between neighbouring roots reduces uptake efficiency and, because it leaches down with water in the soil, plants need deep roots in order to access enough nitrogen. Thus in many ways, the optimal root system phenotypes in soils poor in these nutrients are opposites and when subjected to both these stresses a plant needs to do very different things well in order to access enough nutrients. This is the reason we subject plants to three different nutrient stresses; it is not immediately clear what properties a good

phenotype should have and it is more likely that there are several phenotypes which all perform similarly well under these different stresses.

We set the levels of nutrient stress such that the 'standard' phenotype (which is the default maize phenotype available in the OpenSimRoot repository) would see a reduction of approximately 50% for each nutrient stress as compared to the unstressed case. Figure 5.6 shows the shoot dry weight for different topsoil concentrations of nitrogen, phosphorus and potassium. Based on this, we set the soil nitrogen concentration to $15.5\frac{kg}{ha}$, the soil phosphorus concentration to 0.471 $\frac{\text{kg}}{\text{ha}}$ and the soil potassium concentration to 2.04 $\frac{\text{kg}}{\text{ha}}$.

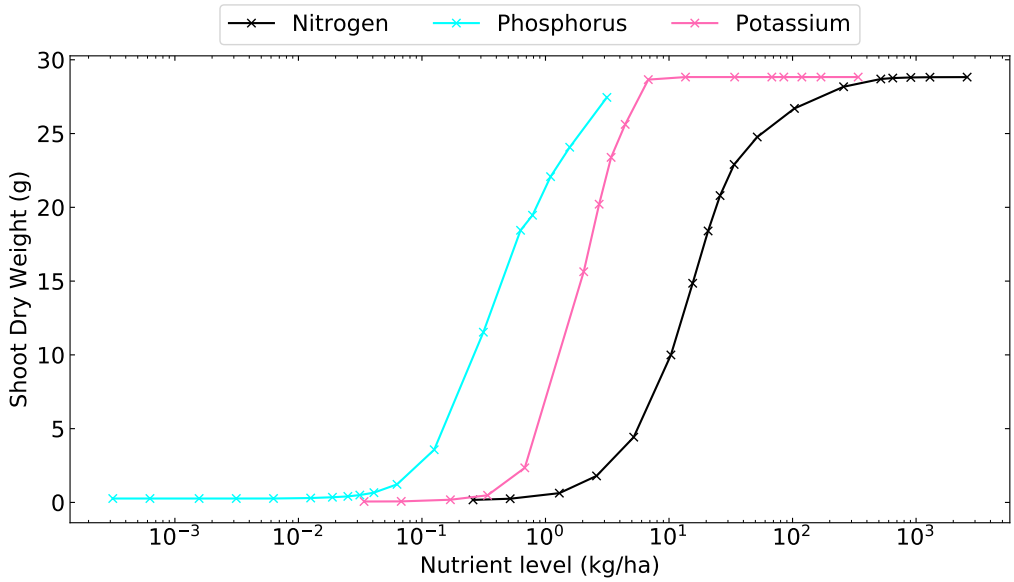


Figure 5.6: Maize shoot dry weight versus topsoil nutrient concentration for nitrogen, phosphrus and potassium. Note that we refer to the concentration of the element in question, where sometimes the concentration of certain molecules containing these elements are used in literature (for example nitrate instead of nitrogen).

Where OpenSimRoot simulation studies usually take into account only a handful of phenes in a factorial design (this means a number of values are specified for each phene and all combinations of different phenes are simulated), we will optimise over 17 different root properties. For context, even with just 2 different values per phene, this would require $2^{17} = 131072$ simulations in a factorial design, not counting repetitions. With 3 or 4 different values per phene, $3^{17} \approx 1.3 \cdot 10^8$ or $4^{17} \approx 1.7 \cdot 10^{10}$ simulations would be needed. Multiply this with at least 4 hours per simulation (usually more) and you would require a fairly sizable computing

facility to work on this for several months, if not years. In other words, optimising over such a high-dimensional parameter space by brute force is not feasible. We will not just optimise over a high-dimensional parameter space, but will also allow values to vary continuously between the lower and upper bounds for each of the phenes we include. As we will see later, we need less than 10000 simulations (including 10 repetitions for each set of inputs) to find good inputs in this high-dimensional parameter space. The root properties and their associated minimum and maximum values, as well as the scaling are shown in Table 5.1. Note that some these phenes correspond to multiple numerical values. For example, different orders of nodal roots have slightly varying emergence angles. For phenes like this, we vary them in the same way and list a representative value in the table. Table B.6 lists some literature values on the basis of which these bounds were chosen.

We do all the relevant calculations on parameter space by rescaling the ranges to the interval $[-1, 1]$. This is both to make programming the necessary algorithms easier and because it adds greater distinction to the dynamics of linear and quadratic terms (which would be very similar if we chose the interval $[0, 1]$). What we then mean by the scaling is how we transform the parameter range to the interval $[-1, 1]$. More specifically, the value of a parameter p with a logarithmic scaling is given by

$$p = p_{min} \left(\frac{p_{max}}{p_{min}} \right)^{\frac{v+1}{2}} \tag{5.6.1}$$

where p_{min} is the minimum value for that parameter, p_{max} the maximum value and v the value in the $[-1, 1]$ we are transforming. For parameters like branching density, that can vary over several orders of magnitude, this logarithmic scaling makes more sense because the effects of changing the branching density from 5 to 10 $\frac{\text{branches}}{\text{cm}}$ will be more like changing the branching density from 50 to 100 $\frac{\text{branches}}{\text{cm}}$ than changing the branching density from 95 to 100 $\frac{\text{branches}}{\text{cm}}$ would.

5.6.1 Wave 1

We created a hypercube with 200 points for our first wave of simulations. This number was chosen based on the computational resources we had available. Each

#	Phene	Abbreviation	Min	Max	Unit	Scaling
1	Seminal root number	SRN	0	15	-	Log
2	Nodal root number	NRN	0	120	-	Log
3	Primary root branching density	PLBF	0.3	30	$\frac{\text{branches}}{\text{cm}}$	Log
4	Seminal root branching density	SLBF	0.4	40	$\frac{\text{branches}}{\text{cm}}$	Log
5	Nodal root branching density	NLBF	0.5	50	$\frac{\text{branches}}{\text{cm}}$	Log
6	Brace root branching density	BLBF	0.4	40	$\frac{\text{branches}}{\text{cm}}$	Log
7	Crown lateral root branching density	CLBF	0.3	30	$\frac{\text{branches}}{\text{cm}}$	Log
8	Lateral root branching density	LLBF	0.3	30	$\frac{\text{branches}}{\text{cm}}$	Log
9	Nodal root emergence time	NRT	4	14	day	Lin
10	Axial root angle	ARA	0	70	degrees	Lin
11	Lateral root branching angle	LRA	60	120	degrees	Lin
12	Fine lateral root branching angle	FLRA	60	120	degrees	Lin
13	Aerenchyma formation	AF	0	0.4	% root volume	Lin
14	Crown lateral root length	CRL	9.2	23.6	cm	Lin
15	Lateral root length	LRL	3.4	7.6	cm	Lin
16	Fine lateral root length	FLRL	0.27	1.84	cm	Lin
17	Major root gravitropism	MAG	0.001	0.1	-	Log

Table 5.1: The 17 root system phenes we include in our optimisation and the abbreviation we will use, the minimum and maximum values (a representative value is chosen for phenes corresponding to different values across different root classes), unit and the scaling. The axial root angle is with respect to the down direction while the lateral and fine lateral branching angles are with respect to the parent root.

simulation was repeated 10 times with different seeds for the random number generator (RNG). Not all simulations completed successfully, those that failed

were retried with different seeds for the random number generator a few times. After this, there were still 83 (out of 2000) combinations of inputs and RNG seed that had not completed successfully. This can happen for a variety of reasons, the most common being a crash in the water model (the stability improvements from Section 4.8 were not present in the code yet). There was only 1 set of inputs that did not succeed for any RNG seed, this was assigned a value of 0, marking it as 'bad'. The mean shoot dry weight could be calculated for all other inputs, though for some this was done with less than 10 repetitions. While not ideal, the failed runs were spread over all input locations so in many cases the mean shoot dry weight was calculated from 8 or 9 repetitions instead of 10.

Optimising the regression for BIC, the following inputs were selected: `1, x1, x2, x3, x10, x16, x17, x1x1, x1x3, x1x4, x1x13, x1x17, x2x5, x2x8, x2x9, x2x12, x2x16, x3x3, x4x4, x4x5, x5x5, x5x9, x10x10, x10x16, x14x15, x17x17.` Here `x1` refers to the first phene in Table 5.1, that is, seminal root number, and so on. As active inputs we choose only those which have sufficient explanatory power by themselves. This means inputs which appear as linear or quadratic terms in the above list, so we don't include `x9`, since it only appears in crossterms. This leaves us with `x1, x2, x3, x4, x5, x10, x16, x17` as active inputs, 8 out of the 17 initial inputs. Then we optimise a degree 2 polynomial regression with respect to the BIC again, forcing all linear terms to be included, which results in the following set of inputs:

`1, x1, x2, x3, x4, x5, x10, x16, x17, x1x1, x1x3, x1x4, x1x17, x2x2, x2x5, x2x16, x3x3, x4x4, x4x5, x5x5, x10x10, x10x16, x17x17.`

After this, the hyperparameters of the Gaussian process, which are the variance, the noise variance and a correlation length for each of the 8 active inputs, were numerically optimised with respect to the log likelihood several times, starting from a number of different values. We need to choose a set of hyperparameters that accurately captures the features of our data. However, with a limited number of points it's often difficult to tell whether a certain set is more appropriate than another. The set we will use, with the highest log likelihood is listed in Table 5.2.

Hyperparameter	Value
Standard deviation	2.03
Noise standard deviation	0.636
x1 correlation length	0.01
x2 correlation length	1.70
x3 correlation length	0.671
x4 correlation length	0.292
x5 correlation length	2.22
x10 correlation length	100
x16 correlation length	100
x17 correlation length	0.268

Table 5.2: Emulator standard deviation, noise standard deviation and correlation lengths for the wave 1 emulator, optimised with respect to the Gaussian process log likelihood.

We note that the first correlation length is the minimum value we allow it to be and the 6th and 7th are the maximum value we have allowed. This indicates that these inputs do not have a big influence on the behaviour of the Gaussian process. This might be because the dependence of shoot dry weight on these inputs already gets captured by the polynomial regression part of the emulator or that the data makes it difficult to discern trends. For example, if the first input is very similar for two data points with very different shoot dry weights, the Gaussian process might conclude that the correlation length in input 1 is very small, while this could also happen due to the fact that the noise is bigger than we think. To check if these hyperparameters really makes sense we run a number of different diagnostics before we continue with the second wave of simulations. First we run leave-one-out diagnostics and see if the proportions of points more than $\sigma, 2\sigma$ and 3σ away from the predicted mean looks reasonable. The leave-one-out diagnostics are listed in Table 5.3.

Now that we have picked hyperparameters, we can generate predictions. We will generate predictions for ten million random points in input space. With these, we can estimate for a given threshold T what the interesting fraction of space P_I is. Remember that, given T, it is defined as

$$P_I := \{\vec{x} \in P | \mu(\vec{x}) + 3\sigma(\vec{x}) \geq T\}. \tag{5.6.2}$$

Threshold	Fraction outside threshold	Expected fraction outside treshold
σ	0.33	0.31372
2σ	0.03	0.0455
3σ	0.01	0.0027

Table 5.3: Leave-one-out diagnostics for the wave 1 emulator showing the fraction of wave 1 results that fall outside the given thresholds. See Section 5.4 for an explanation of leave-one-out diagnostics.

The highest mean shoot dry weight in our training data is equal to 20.07 grams, we set the threshold for cutoff at 95% of this, so equal to 19.06 grams. Now that we set a threshold we will determine the best locations for 20-day simulations, as described in Section 5.5. First we choose a set X of 1000 reference points that we use to calculate

$$\nu(\vec{z}) := \sum_{\vec{x} \in X} \sigma_{\vec{z}}^2(\vec{x}) \omega(\vec{x}), \tag{5.6.3}$$

for any prospective points $\vec{z}$. We choose these thousand points by calculating

$$\omega(\vec{x}) = e^{-\frac{(T-(\mu(\vec{x})+3\sigma(\vec{x})))^2}{\sigma^2(\vec{x})}}, \tag{5.6.4}$$

for every $\vec{x}$ in a set of ten million predictions. Then we choose the points with the highest $\omega(\vec{x})$. If these reference points are too close together, then the 20-day simulations will be optimised to decrease the variance in a small region of parameter space. So we impose a minimum distance between the reference points. The minimum distance between reference points was calculated by generating 100 hypercubes consisting of 1000 points each and calculating the average minimum distance between points, which was equal to 1.80. We multiplied this by 0.8 and imposed this as minimum distance between the reference points.

5.6.2 Wave 1.5

With these reference points calculated, we choose 500 points as inputs for 20-day simulations, with 10 repetitions each. As explained in Section 5.5, we choose points that minimise the predicted variances for the points in the reference set, weighted by $\omega(\vec{x})$. The results of these 20-day simulations can be converted to predictions for the shoot dry weight at 42 days by using our initial 1917 successfully completed 42-day simulations to fit a second order polynomial regression,

the forward regression, over a selection of 25 OpenSimRoot outputs. These outputs include shoot dry weight, measures of the size of the root system such as the root length of different root classes and measures of nutrient uptake, see Table B.7 for the complete list. Figure 5.4 shows the relationship between the above 20-day values and the shoot dry weight at 42 days. With these we parametrised a second order polynomial regression which has an adjusted coefficient of determination (r-squared) of 0.942.

4973 out of 5000 (500 inputs times 10 repetitions) 20-day simulations completed successfully. These give us give 4973 predicted shoot biomasses at 42 days, which are averaged for each set of inputs and then added to the 200 mean shoot biomasses from the first wave. This gives us 700 datapoints which we use to fit a wave 1.5 regression and Gaussian process. The hyperparameters for the wave 1.5 Gaussian process are listed in Table 5.4 and the corresponding leave-one-out diagnostics are listed in Table 5.5.

Hyperparameter	Value
Variance	2.42
Noise variance	1.91
x1 correlation length	0.439
x2 correlation length	0.0213
x3 correlation length	0.197
x4 correlation length	0.660
x5 correlation length	0.360
x10 correlation length	100
x16 correlation length	100
x17 correlation length	0.142

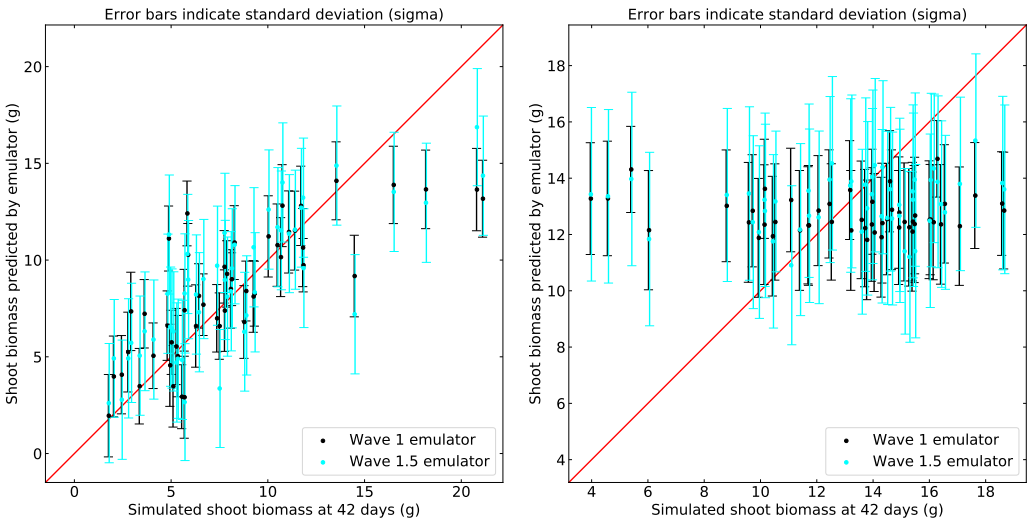
Table 5.4: Emulator standard deviation, noise standard deviation and correlation lengths for the wave 1.5 emulator, optimised with respect to the Gaussian process log likelihood.

The fraction outside 1σ is smaller than we would expect if errors were distributed normally. It is also smaller than it was for wave 1 while the fractions outside 2 and 3 σ are similar to those of wave 1. This is likely due to the higher variance and noise variance, which increases σ, making it more unlikely for values to fall outside the $[\mu - \sigma, \mu + \sigma]$ interval. To make sure that the Gaussian process works well we ran some simulations to generate data to validate against. We created

Threshold	Fraction outside threshold	Expected fraction outside treshold
σ	0.25	0.31372
2σ	0.034	0.0455
3σ	0.0071	0.0027

Table 5.5: Leave-one-out diagnostics for the wave 1.5 emulator showing the fraction of results that fall outside the given thresholds. See Section 5.4 for an explanation of leave-one-out diagnostics.

two sets of 50 inputs, each with 10 repetitions. The first set is a Latin hypercube over all of parameter space. The second set is a selection of 50 points on the boundary of the interesting region. More specifically, they are the points with the highest weight $\omega(\vec{x})$ as defined in equation 5.5.1, selected from the set of points that was used to determine the optimal locations for the wave 1.5 simulations. Figures 5.7a and 5.7b show the predicted means and standard deviations versus the actual values for the wave 1 and wave 1.5 emulators. The mean error of the wave 1 emulator is 1.97 grams for the first validation set and 2.84 grams for the boundary validation set. For the wave 1.5 emulator these mean errors are 2.24 and 2.58 grams, respectively.



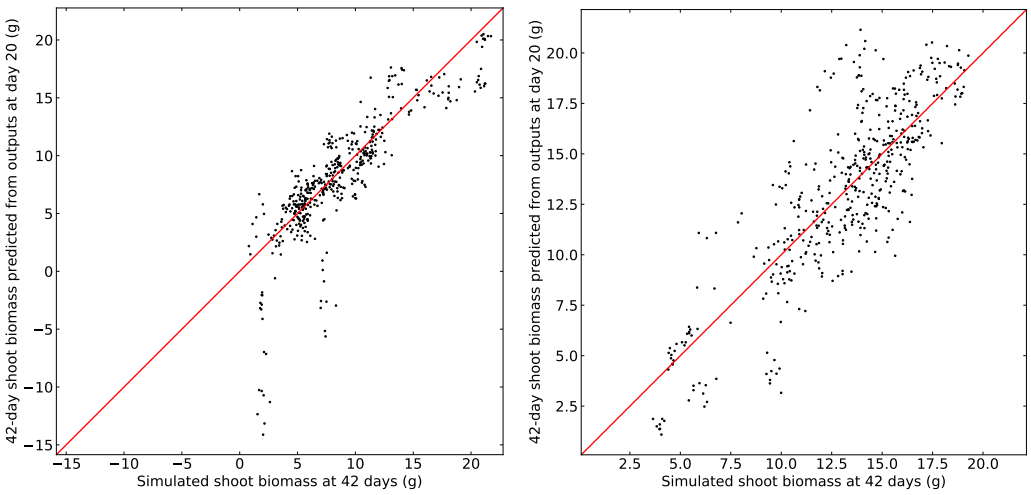
(a) Validation points distributed across parameter space.

(b) Validation points in boundary region.

Figure 5.7: Emulator predicted shoot biomass at 42 days versus shoot biomass simulated by OpenSimRoot at 42 days for the wave 1 and wave 1.5 emulator. The bars indicate the emulator standard deviation.

It is surprising that the wave 1.5 emulator actually performs worse (that is, the mean errors are bigger) on the first validation set than the wave 1 emulator,

considering it has more information. It did perform a little bit better on the boundary validation set, but since the boundary validation points were taken from the set of reference points that was used to decide where the wave 1.5 simulations should be done, this improvement is quite small. This warranted further investigation. Figures 5.8a and 5.8b show the 42-day shoot dry weight predicted by the forward regression model versus the actual 42-day shoot dry weight.



(a) Validation points distributed across parameter space

(b) Validation points in boundary region

Figure 5.8: Shoot biomass at 42 days as predicted by the forward regression, a polynomial regression of degree two on selected outputs at day 20, versus shoot biomass simulated by OpenSimRoot at 42 days.

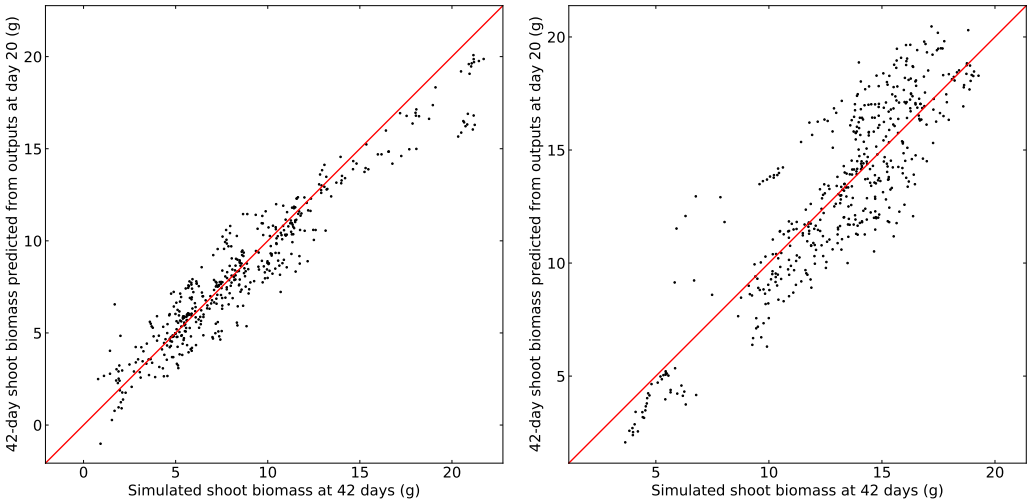
As we can see in Figure 5.8a, the forward regression model predicts negative shoot dry weights in about 20 instances, which of course is not physically possible. Feeding these unphysical values into the Gaussian process will in turn make it less accurate. A common way to ensure a regression model predicts positive values is to make it predict the logarithm of the relevant value. This is done by taking the logarithm of the training data y-values, training the regression model on that and taking the exponent of the predicted values. However, because we are using least squares to determine the regression coefficients, this will cause small values to weigh more heavily. Because we are trying to optimise for shoot dry weight, we care more about accuracy at high values than at low values; we will discard the lowest values and even if the error there is relatively large, they are unlikely to be among the highest shoot dry weights. So instead we will regard any predicted 42-day shoot dry weight below 0 as 0 before using it to train a Gaussian

process. This improves the wave 1.5 emulator average error from 2.24 to 1.79 on the first validation set and from 2.58 to 1.93 on the boundary validation set.

The large bounds we put on correlation lengths mean that some of our inputs are not doing much to inform the Gaussian process, which is unexpected because we selected the inputs with the most explanatory power. So we try restricting the correlation lengths to the interval $[0.2, 5]$ instead of $[0.01, 100]$ for both the wave 1 and wave 1.5 emulator. With this and with restricting the 42-day shoot dry weight as predicted by the forward regression to positive values, the mean error of the wave 1 emulator is 2.16 g on the first validation set and 2.61 g on the boundary validation set, for the wave 1.5 emulator these values were 1.78 g and 1.95 g, respectively.

Finally we should note that we are using all 20-day outputs in the forward regression model to predict forward to the shoot dry weight at 42 days. But since we used the Bayes' information criterion to select active inputs for the regression term in our Gaussian process, it makes sense to also try to apply that to the forward regression model. We first constructed a linear model and optimised for BIC to select which inputs we include. This led to a selection of 17 out of the 25 inputs, see Table B.8. Then we constructed a regression of degree 2 and optimised the active inputs using BIC. This reduced the mean error on the first validation set from 1.86 g to 1.06 g and reduced the mean error on the second validation set from 1.79 g to 1.52 g. Figure 5.9 shows that the forward regression model is not only predicting the shoot dry weight at 42 days with higher accuracy but also predicts fewer negative values as compared to the predictions in Figure 5.8. This shows that, perhaps counterintuitively, reducing the amount of variables considered can improve accuracy, and underscores that one should take care not to overfit.

After this fine-tuning, we see in Figure 5.10 that the wave 1.5 emulator is an improvement on the wave 1 emulator. Most of the wave 1.5 predictions are closer to the real values than the wave 1 predictions. In particular, the wave 1.5 emulator improves on the wave 1 emulator for the 3 points with the largest simulated shoot biomasses in Figure 5.10a and the 4 points with the smallest simulated

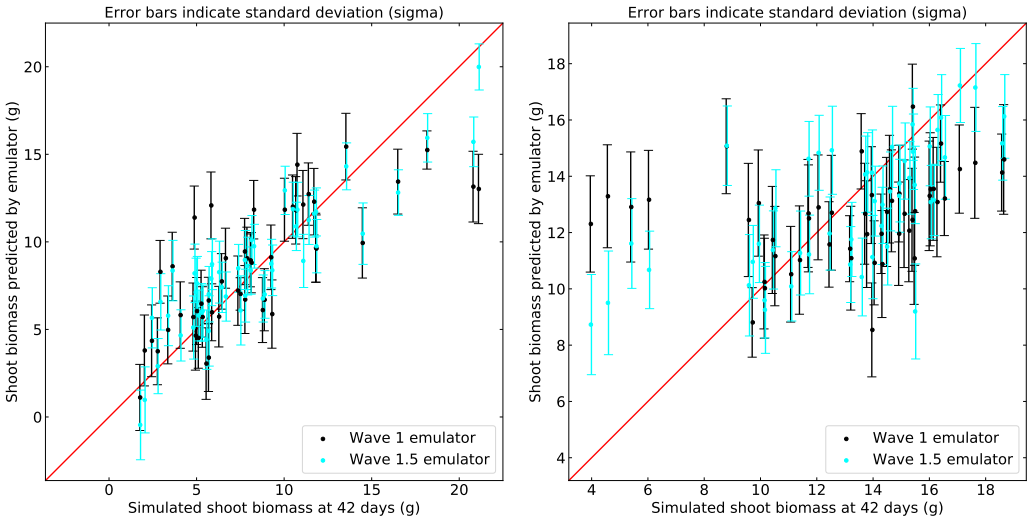


(a) Validation points distributed across parameter space

(b) Validation points in boundary region

Figure 5.9: Shoot biomass at 42 days as predicted by the forward regression, a polynomial regression of degree to on selected outputs at day 20, versus shoot biomass simulated by OpenSimRoot at 42 days, after using the Bayes information criterion to select which inputs to include in this regression.

shoot biomasses in Figure 5.10b, which are the points with the largest errors.



(a) Validation points distributed across parameter space

(b) Validation points in boundary region

Figure 5.10: Emulator predicted shoot biomass at 42 days versus shoot biomass simulated by OpenSimRoot at 42 days for the wave 1 and wave 1.5 emulator after improving the shoot dry weight predictions made based on the 20-day simulations which the wave 1.5 emulator uses. The bars indicate the emulator standard deviation.

Now we are satisfied that our methods are correct we construct our final wave 1.5 emulator by adding the validation data to the training data. By adding the two

sets of validation data in one by one we verified that this improves, or at the very least does not reduce, accuracy on the other validation set. With this final wave 1.5 emulator we will construct a set of inputs for the second wave of full 42-day simulations. The maximum mean shoot dry weight in the data on which this final wave 1.5 emulator is trained is 21.12 g so we set the wave 1.5 threshold, $T_{1.5}$, at which we consider a point 'interesting' equal to $T_{1.5} = 0.95 \cdot 21.12 = 20.06$ g. Any point $\vec{x}$ for which $\mu(\vec{x}) + 3\sigma(\vec{x}) < T_{1.5}$ will be excluded from future consideration. Under the assumption that our errors are distributed unimodally, the Pukelsheim 3σ rule [163] tells us that we will at most exclude 2.5% of parameter space from consideration in error with this choice of 3σ. This is because for unimodal distributions less than 5% will fall outside 3σ and half of these will fall outside 3σ on the low end of the distribution, which we do want to include. For a normal distribution, only about 0.3% of samples falls outside 3σ so we would mark at most 0.15% of parameter space uninteresting in error. Since we set our threshold $T_{1.5} = 20.06$ g and not at the maximum observed value, we are even less likely to exclude good inputs from consideration.

Since the parameter space we are working in is 17-dimensional, it is difficult to visualise the predictions of the Gaussian processes. By projecting down onto pairs of two dimensions we can get an idea of the shape of the interesting region defined by our threshold and emulators and how this changes between the wave 1 and wave 1.5 emulator as visualised in figures 5.11 and 5.12. We see some differences in the interesting regions, but the two emulators seem to agree on the location of the bulk.

Instead of plotting the optical depth, which is the size of the hypervolume projected down onto the relevant 2 dimensions, and the average of the mean shoot dry weights, we also plot the minimum and maximum predicted mean shoot dry weights in Figure 5.13. This gives us an idea of the range of predictions and where the emulator expects the highest values to be. We see that the minimum values range from slightly less than 14 to 15.4 g. It is not surprising the range does not extend further down because going any lower would mean a point falls below the threshold, even with maximum variance. The highest minimum values look like they are at the edges of the interesting set. This is likely due to the fact

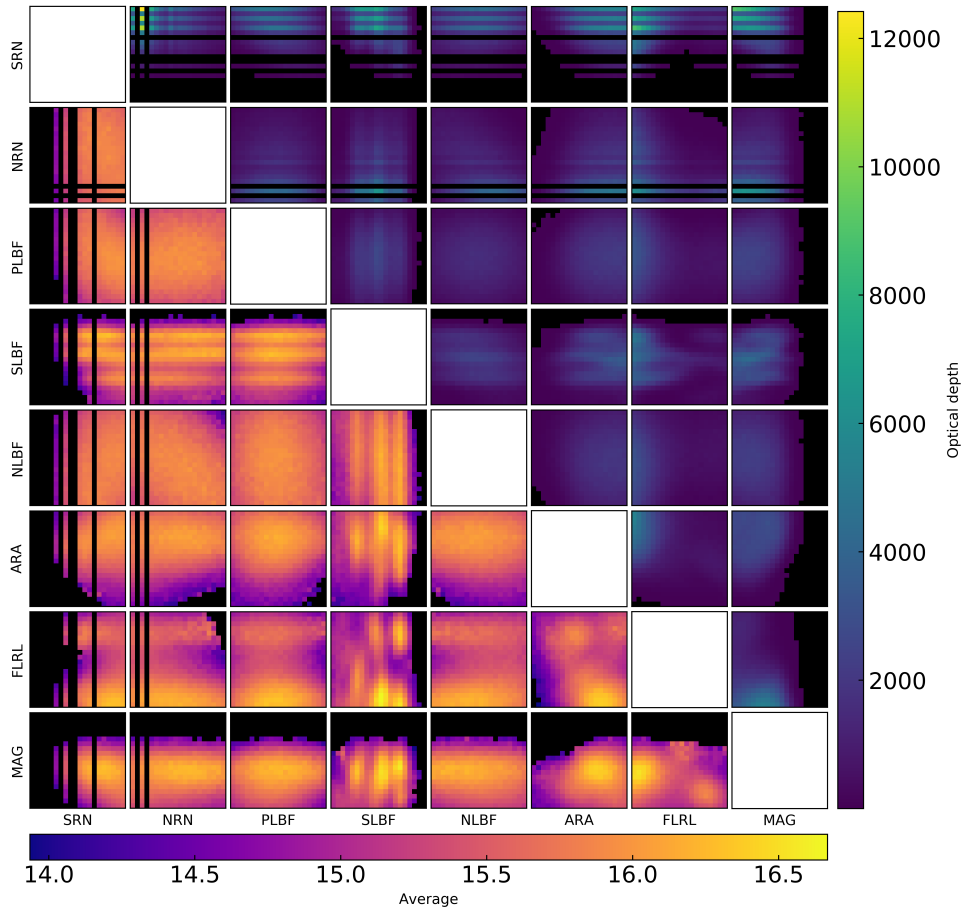


Figure 5.11: **An overview of the predictions of the wave 1 emulator of maize shoot dry weight after 42 days.** Each panel shows the projection onto two inputs (dimensions) of the average of emulator predictions for points in the interesting region, for panels below the diagonal, or the optical depth, which is the number of points in the interesting region, for panels above the diagonal, with each pixel representing a number of points for which the values in the two dimensions in question match the x, y coordinates in the panel in question. Black regions contain no points in the interesting region (or, in the case of the first two inputs, which are discrete, are not part of parameter space). Only the 8 active inputs are displayed.

that there are fewer points in the interesting set in the hyperslices of parameter space represented by the relevant pixels. The plots with maximum mean shoot dry weight seem to show a peak in the middle of the interesting set with values decreasing towards the edges. For some parameters there is a clear preference and parts of parameter space are completely excluded. For example, low seminal root numbers (SRN) do not appear in the interesting set at all.

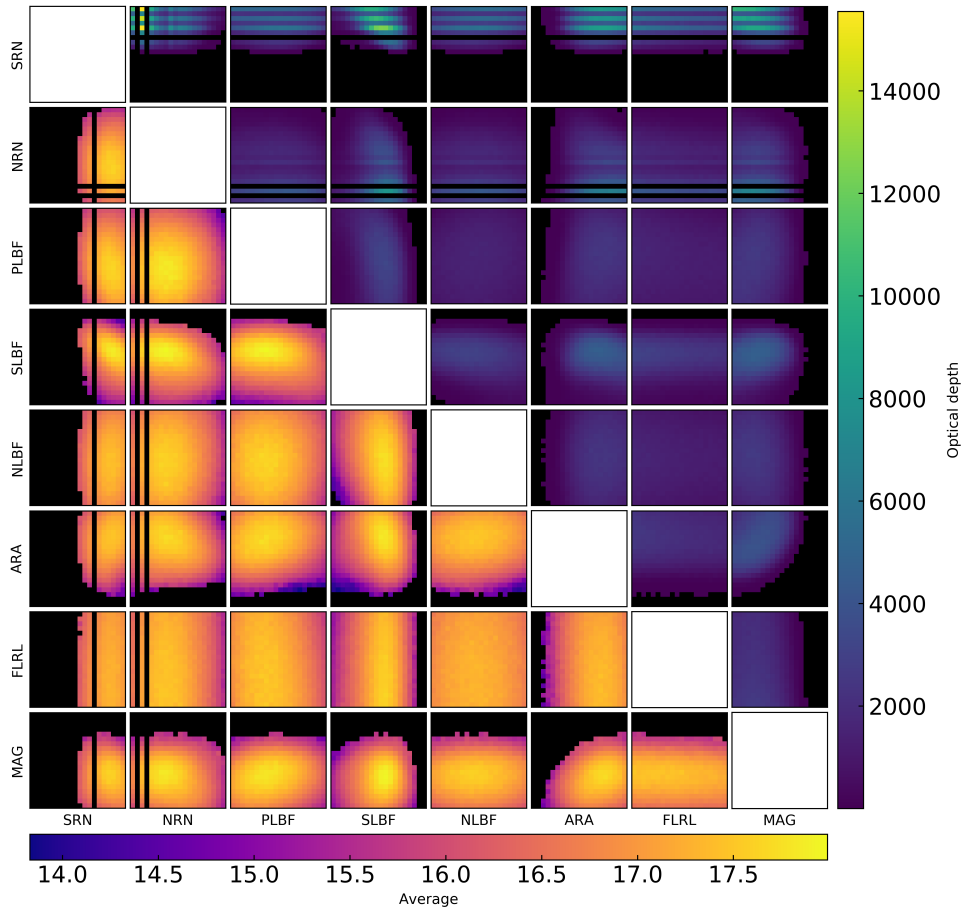


Figure 5.12: **An overview of the predictions of the wave 1.5 emulator of maize shoot dry weight after 42 days.** Each panel shows the projection onto two inputs (dimensions) of the average of emulator predictions for points in the interesting region, for panels below the diagonal, or the optical depth, which is the number of points in the interesting region, for panels above the diagonal, with each pixel representing a number of points for which the values in the two dimensions in question match the x, y coordinates in the panel in question. Black regions contain no points in the interesting region (or, in the case of the first two inputs, which are discrete, are not part of parameter space). Only the 8 active inputs are displayed.

5.6.3 Wave 2

We constructed the first wave of simulations as a Latin hypercube in order to have an even coverage of parameter space. For the wave 2 simulations we similarly want to cover the interesting part of parameter space, as determined by our wave 1.5 emulator, as uniformly as possible. To achieve this, we generate a large number of Latin hypercubes on the entire parameter space and keep only the points in the hypercube that fall in the interesting region. From the resulting reduced hypercubes that contain approximately the target amount of points we pick the one that has the largest minimum distance between points because

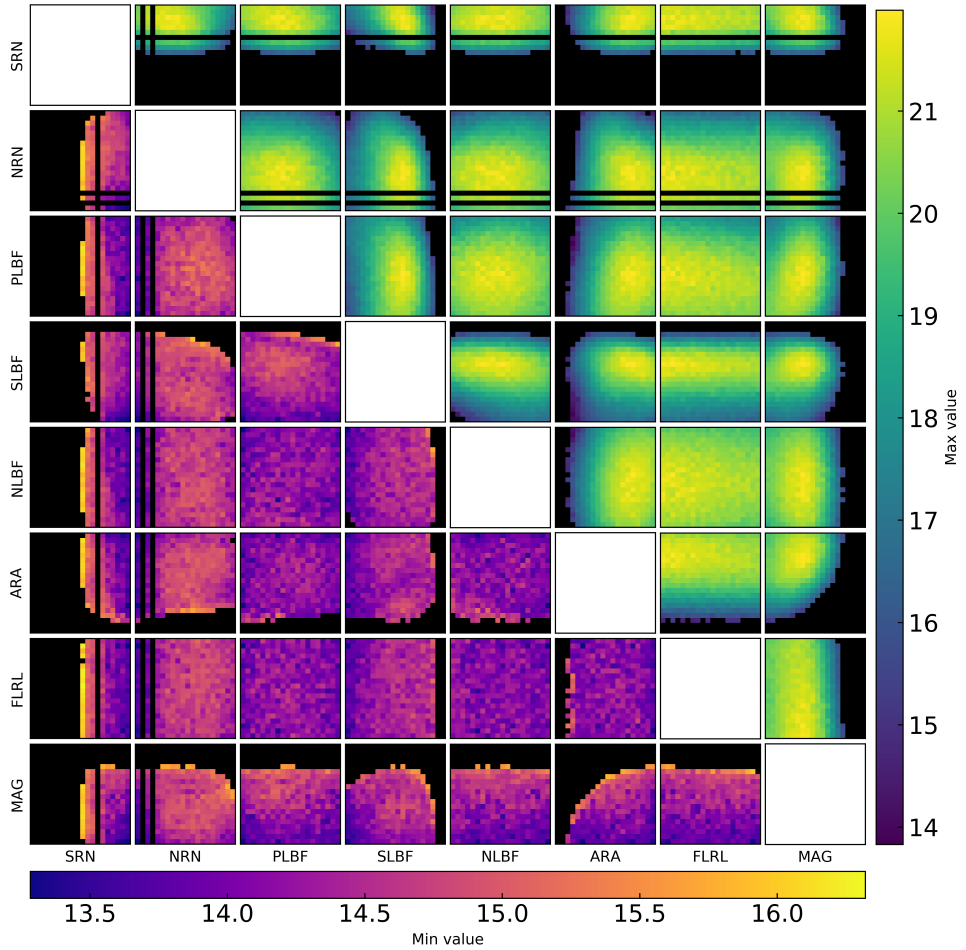


Figure 5.13: **An overview of the predictions of the wave 1.5 emulator of maize shoot dry weight after 42 days.** Each panel shows the projection onto two inputs (dimensions) of the minimum of points in the interesting region (for panels below the diagonal) or the maximum of points in the interesting region (above the diagonal), with each pixel representing a number of points for which the values in the two dimensions in question match the x, y coordinates in the panel in question. Black regions contain no points in the interesting region (or, in the case of the first two inputs, which are discrete, are not part of parameter space). Only the 8 active inputs are displayed.

that one is most likely to evenly cover the interesting region of parameter space. While it's not guaranteed that the resulting set of points is evenly spaced and provides good coverage of the interesting set, it will almost certainly be better than a set of randomly chosen points. We also added the point for which the wave 1.5 emulator predicted the highest mean shoot dry weight, which was 21.99 g to the wave 2 inputs.

The highest mean shoot dry weight in the wave 2 data was 23.03 g (this was not the point the wave 1.5 emulator expected to be the highest), a clear improvement

over the wave 1 and wave 1.5 mean shoot dry weights. Figure 5.14 shows the shoot dry weight for the wave 2 inputs as predicted by the wave 1.5 emulator versus the values as simulated by OpenSimRoot. It is clear that the emulator is not very accurate but our main cause for concern is if it underestimated values by more than 3σ because this would mean we could be excluding parts of parameter space from consideration in error. This only is the case for a few points, the majority falls within 3σ of the predicted value.

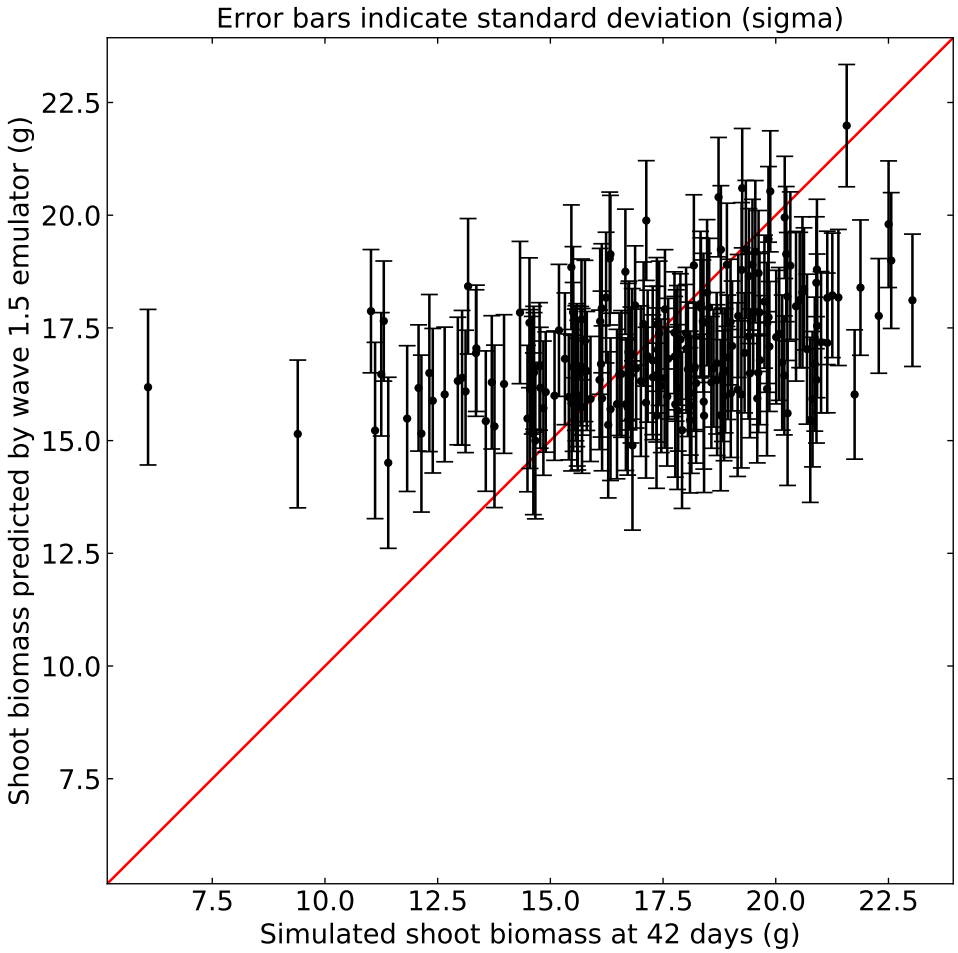


Figure 5.14: Wave 2 results versus emulator wave 1.5 prediction.

When we construct a wave 2 emulator we can use just the wave 2 data alone or add in the data from wave 1, wave 1.5 and the validation data. Since the wave 2 emulator will only be used to make predictions in a small region of parameter space, the former approach makes sense because it only uses data from the relevant region. However, the latter approach takes more data into account and this can of course make for a better emulator. In addition, if we construct an

emulator from just wave 2, this emulator might not see the effect an input has because we are already 'at the top of the mountain' in that dimension. To find out which approach works better we constructed both emulators and compared them. The leave-one-out diagnostics of both emulators looked similar where we expect them to be, so this does not tell us much. To get some more insight, we made a large number of predictions in the interesting region (as classified by the wave 1.5 emulator) and make a histogram of the predicted standard deviations, as shown in Figure 5.15. All emulators showed a reasonably smooth distribution except for the wave 2 emulator that is only trained on wave 2 data, which showed a very sharp peak at the highest value of its distribution. This means that this emulator is unsure about almost all of its predictions, because most points are far away, relative to the correlation lengths, from the data it was trained on. In this case it is because two of the correlation lengths are very small, which means that unless a point is very close to a training point in either of the two dimensions corresponding to these small correlation lengths, the emulator will make predictions with high uncertainty.

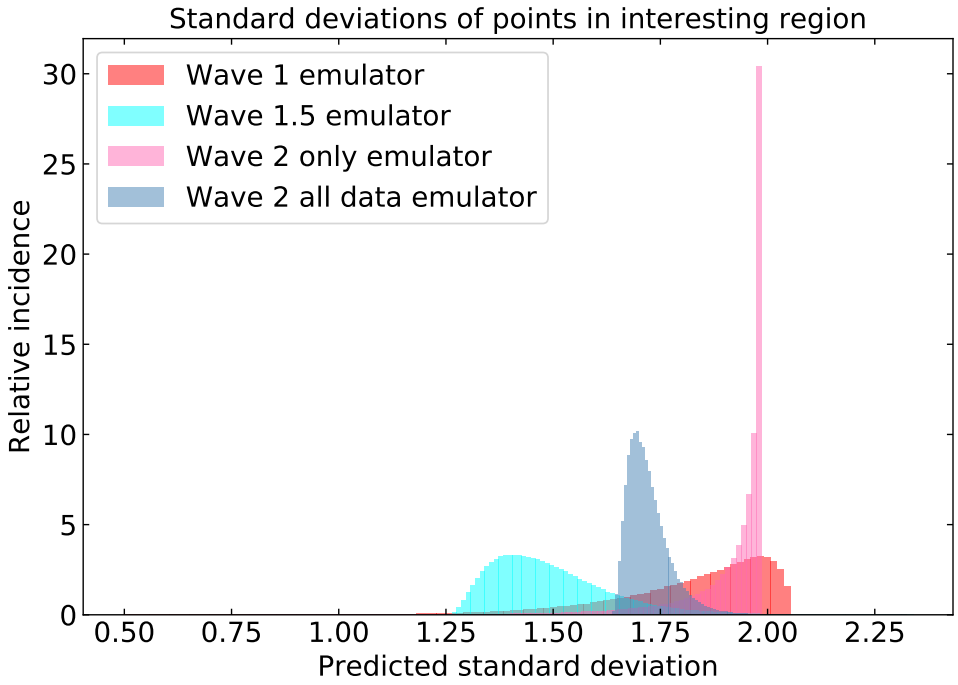


Figure 5.15: A histogram showing the distribution of predicted standard deviations for the wave 1, wave 1.5 and 2 different wave 2 emulators for points in the region of parameter space marked as interesting by the wave 1.5 emulator.

In order to improve the wave 2 emulators we first of all adjust the hyperparam-

eters of these emulators, increasing the lower bounds of the correlation lengths from 0.2 to 0.25. More importantly, we also reviewed which of the 17 root traits that make up our parameter space we train these emulators on. Recall that we used the Bayesian information criterion (BIC) to choose the inputs which explained the most variation and trained the emulators using only this selection of 8 inputs. If these 8 inputs determine the large scale behaviour, it makes sense that once we select the values that maximise shoot dry weight in these 8 inputs, we eventually reach a point where further increases in shoot dry weight can only be achieved by looking at the other 9 inputs. We optimised the regression terms for BIC for just the wave 2 dataset and the dataset containing every wave, and calculate the P-value for all these terms. The P-value indicates the probability that the variation that appears to be explained by a term is actually due to chance. A P-value smaller than 0.05 is generally seen as a good indication that a term should be included. Table B.9 lists the P-values for all relevant regression terms for wave 1, wave 1.5 and the two wave 2 datasets. Based on the terms selected by BIC optimisation and their corresponding P-values we decided to add inputs 8 (lateral root branching density), 9 (nodal root emergence time), 13 (aerenchyma formation) and 15 (lateral root length) as active inputs, bringing the total to 12 out of 17. After retraining the wave 2 emulators and making predictions we again plot the distribution of standard deviations in a histogram, see Figure 5.16. Not only do we now see smooth distributions for all emulators, the wave 2 emulators also have lower standard deviations than the wave 1 and wave 1.5 emulators, which means that the expected accuracy of predictions has increased.

Now we are satisfied with the wave 2 emulator we use it to reasses the interesting set. For the points which the wave 1.5 emulator expects to be above the wave 1.5 threshold of 20.06 g, we make predictions with the wave 2 emulator. Note that we could use the wave 2 emulator to make predictions on the entire parameter space but since all of the wave 2 simulations are in the wave 1.5 interesting set, the wave 2 emulator does not have more information on most of parameter space than the wave 1.5 emulator does. However, since we have observed higher shoot biomasses in wave 2, we increase the threshold to $0.95 \cdot 23.03 = 21.88$ g. The wave 2 emulator expects 1.66715% of parameter space to be interesting, down from the 4.77827% for the wave 1.5 emulator. Figures 5.17 and 5.18 show the

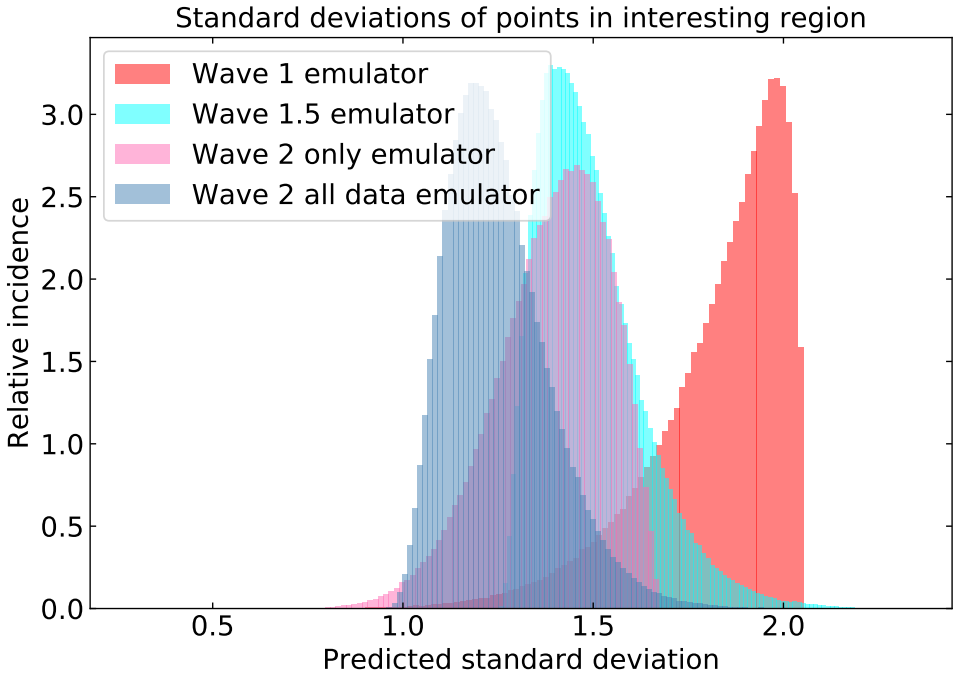


Figure 5.16: A histogram showing the distribution of predicted standard deviations for the wave 1, wave 1.5 and 2 different wave 2 emulators for points in the region of parameter space marked as interesting by the wave 1.5 emulator. This figure is similar to Figure 5.15 except that the wave 2 emulators are now trained on 12 of the 17 inputs instead of the 8 inputs we used before.

two-dimensional projections of the interesting part of parameter space. Because we used more inputs to train the wave 2 emulator more are displayed in these figures than before but they are otherwise very similar to those for wave 1.5. The projections of the interesting set onto some pairs of inputs are smaller, which does not necessarily have to happen even if the interesting region gets smaller, but the overall shape is similar. The minimum and maximum predictions have increased, which makes sense because the wave 2 simulations have higher shoot dry weight on average and the threshold has increased while the emulator variance has gone down.

Now we could repeat what we did for wave 1 and run a large number of 20-day simulations, then predict the 42-day shoot dry weight from those using a forward regression. However we won't do this for two reasons. The first is that we have already excluded 95% of parameter space from consideration so we are looking at a much smaller space. This means that it is a lot easier to get good coverage of the part of parameter space we have left. The second reason is that the procedure becomes less and less useful the closer we get to the highest possible values of the

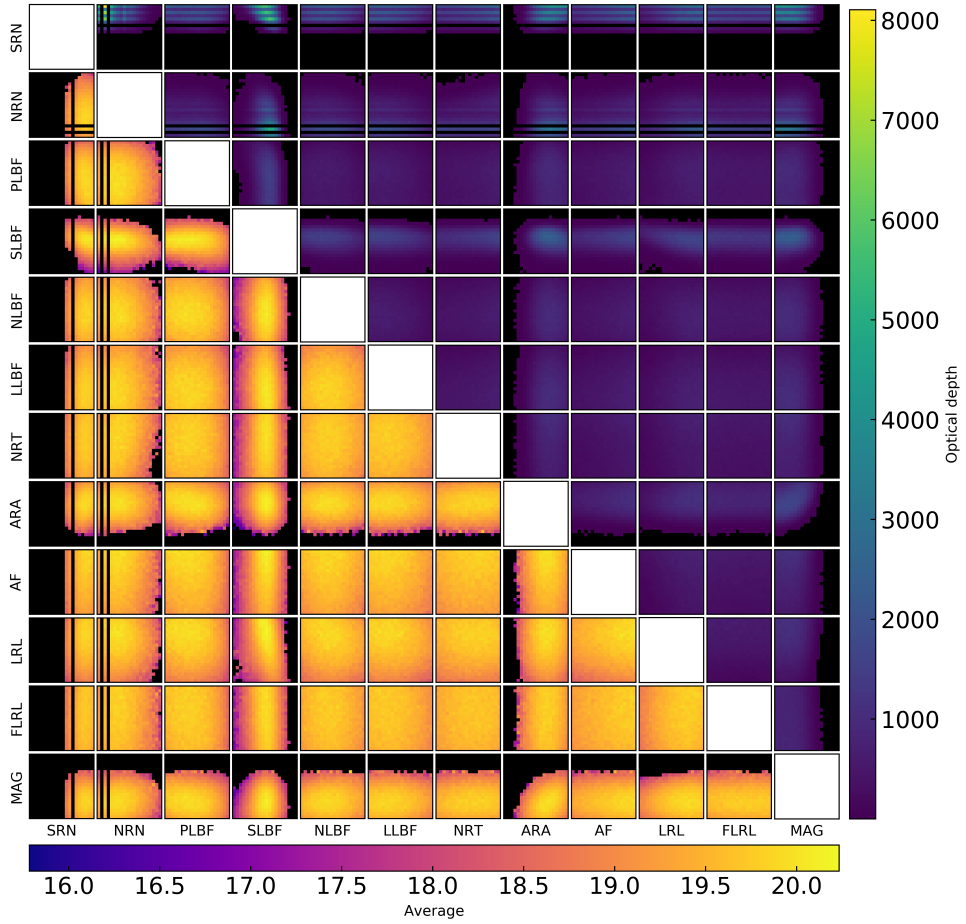


Figure 5.17: **An overview of the predictions of the wave 2 emulator of maize shoot dry weight after 42 days.** Each panel shows the projection onto two inputs (dimensions) of the average of emulator predictions for points in the interesting region, for panels below the diagonal, or the optical depth, which is the number of points in the interesting region, for panels above the diagonal, with each pixel representing a number of points for which the values in the two dimensions in question match the x, y coordinates in the panel in question. Black regions contain no points in the interesting region (or, in the case of the first two inputs, which are discrete, are not part of parameter space). Only the 12 active inputs are displayed.

model output we are maximising. This is because phenotypes near the maximum possible shoot dry weight at day 42 will look very similar in certain respects at day 20. For example, they are unlikely to be nutrient stressed at day 20, because this would mean their growth was already being limited, and their shoot dry weights are likely to be very similar at day 20 because otherwise the positive feedback loop of more carbon availability leading to more nutrient uptake leading to even more shoot growth would translate to big differences in the final 22 days. This was confirmed by the fact that the standard deviation of the residuals of the forward regression was larger when this was applied to wave 2 data, compared to

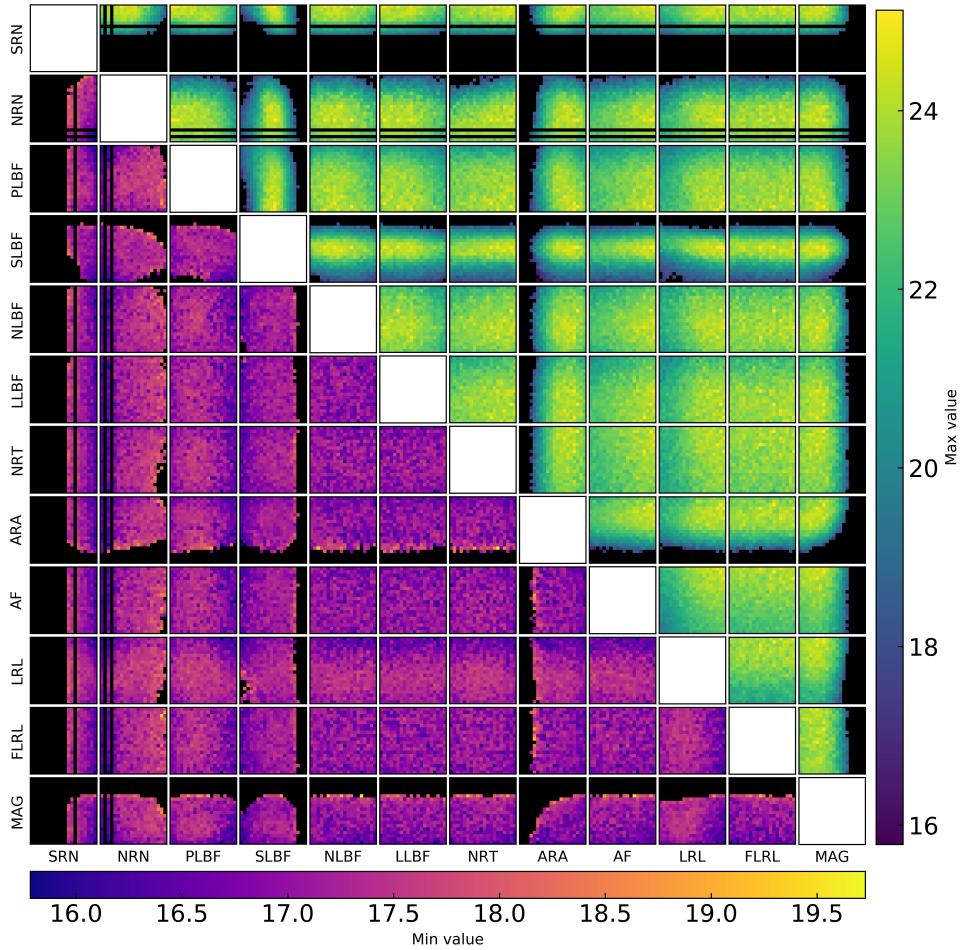


Figure 5.18: **An overview of the predictions of the wave 2 emulator of maize shoot dry weight after 42 days.** Each panel shows the projection onto two inputs (dimensions) of the minimum of points in the interesting region (for panels below the diagonal) or the maximum of points in the interesting region (above the diagonal), with each pixel representing a number of points for which the values in the two dimensions in question match the x, y coordinates in the panel in question. Black regions contain no points in the interesting region (or, in the case of the first two inputs, which are discrete, are not part of parameter space). Only the 12 active inputs are displayed.

when it was applied to wave 1 data. This indicates the size of the errors increased. For this reason we will not use 20-day simulations in any further waves.

5.6.4 Wave 3

For our third and final wave we create a restricted Latin hypercube with approximately 213 points using the wave 2 emulator (because there is randomness involved in creating a Latin hypercube, we can not guarantee an exact number of points will fall in the small fraction of space that is interesting according to the wave 2 emulator so we accept a range of sizes). We add to this the 5 points which

the emulator expects to have the highest mean shoot dry weight. Figure 5.19 shows the values predicted by the wave 2 emulator plotted against the simulated values. As it turns out, 3 of the 5 points which the wave 2 emulator predicted would have the highest mean shoot dry weights did have the 3 highest simulated shoot dry weights, though they were slightly lower than the emulator predicted. The other predictions were reasonably good, with most within 3 standard deviations of the actual value. While the wave 2 emulator predicted some values above $25g$, the highest wave 3 result was just slightly above 24 g.

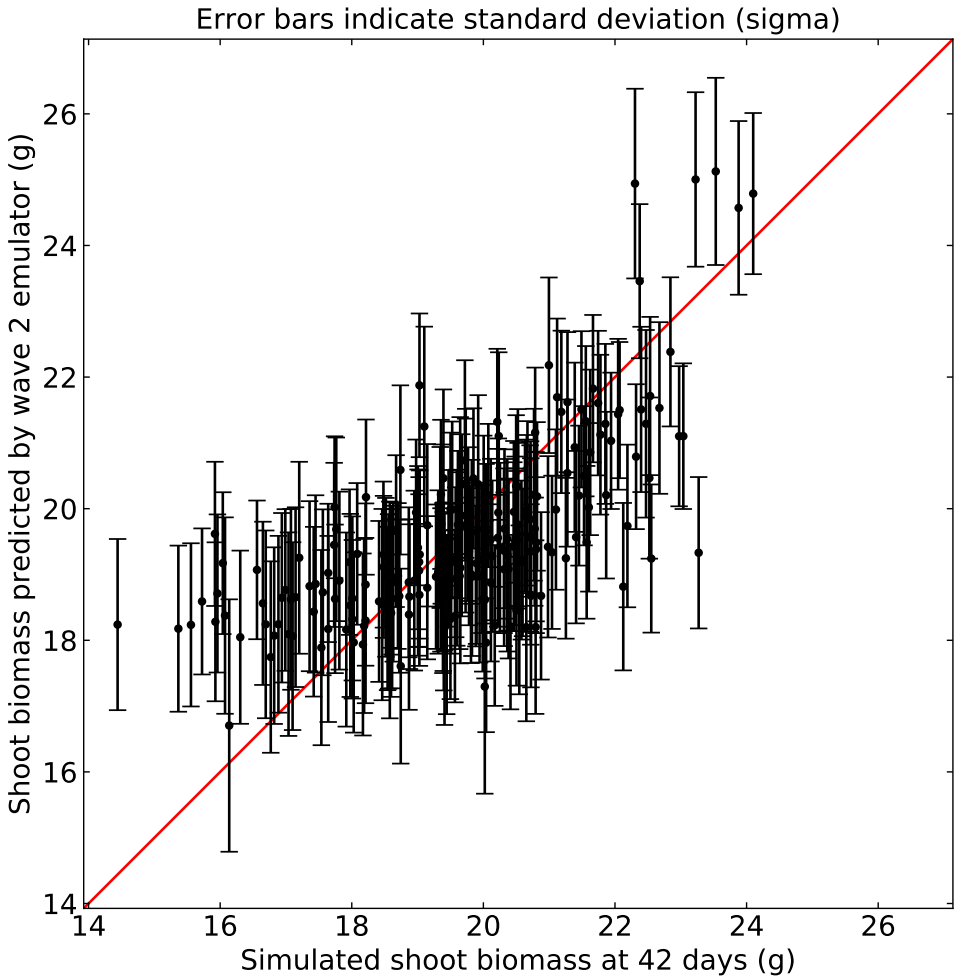


Figure 5.19: Wave 3 results versus wave 2 emulator predictions. The bars indicate the emulator standard deviation. The 5 points with the highest shoot dry weight as predicted by the wave 2 emulator are the 5 points the emulator thought would have the highest shoot dry weight, and they do indeed have among the highest shoot dry weights.

Training a wave 3 emulator, which uses all data so far and is valid on the interesting set according to the wave 2 emulator, we determine another, even smaller

region of interest. But first we make sure that the emulator is still behaving properly. As Figure 5.20 shows, the distribution of predicted standard deviations is smooth and the wave 3 emulator is more certain of its predictions than the wave 2 emulator, as expected.

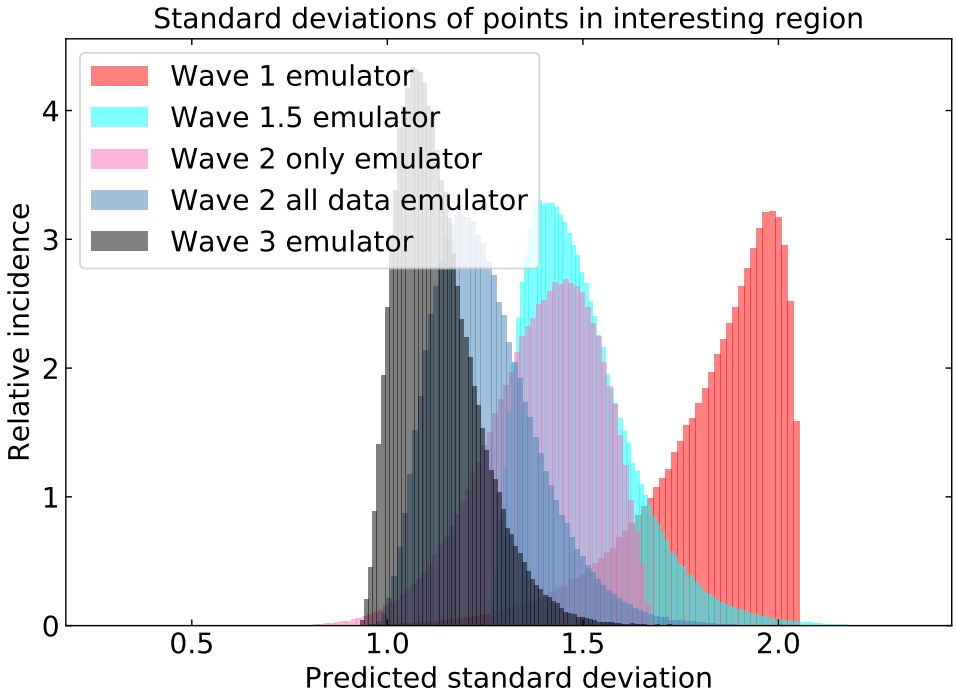


Figure 5.20: A histogram showing the distribution of standard deviations for the wave 1, wave 1.5, wave 2 and wave 3 emulators for points in the relevant interesting regions of parameter space. Note that for the wave 3 emulator, the interesting region is determined by the wave 2 emulator, which is more strict than the wave 1.5 emulator, so this interesting region is smaller.

Figure 5.21 summarises the results of the three waves of simulations we have done, showing the average nutrient uptake, normalised with respect to the nutrient uptake of the reference plant grown in soil with high nutrient availability and the simulated shoot dry weight for every wave as well as the performance of the reference phenotype. Most phenotypes in wave 1 have lower shoot dry weights than the reference phenotype in stressed conditions, the wave 2 phenotypes perform a lot better already while for wave 3 almost all phenotypes have higher simulated shoot dry weights than the reference phenotype in stressed conditions. The average shoot dry weight increases from 8.30 g in wave 1 through 17.38 g in wave 2 to 19.63 g in wave 3. The highest shoot dry weight value in wave 3, growing in stressed conditions, is 24.10 g, only 4.73 g below the reference plant

growing in unstressed conditions.

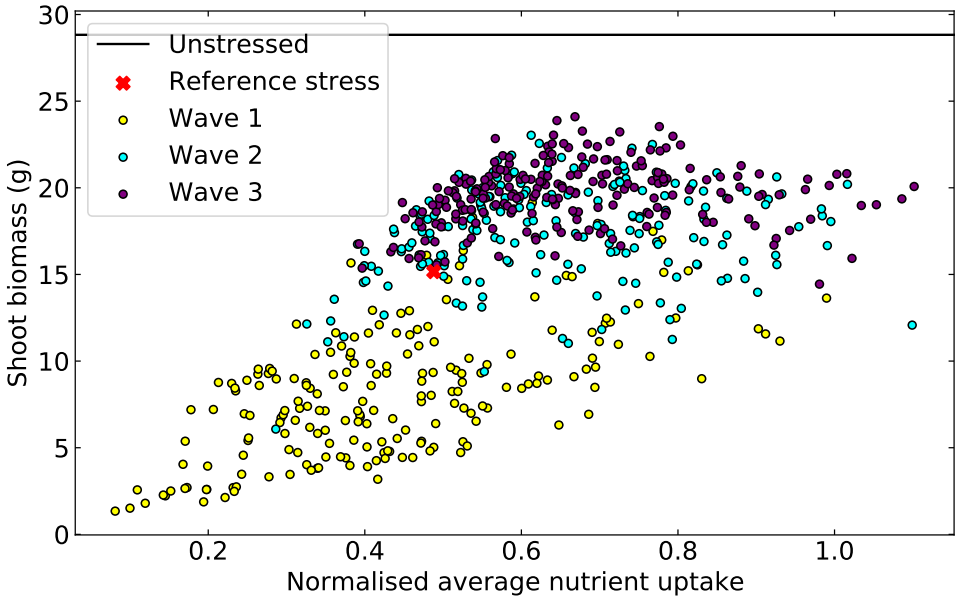


Figure 5.21: An overview of the results from all 3 waves of simulations. The horizontal axis shows the average nutrient uptake, normalised with respect to the nutrient uptake of the plants grown in soil with high nutrient availability. The simulated shoot dry weight is on the vertical axis. Each wave is displayed in a different colour, the reference unstressed shoot dry weight is shown by the black line and the reference phenotype under stress is shown in red (around 0.5, 15).

Can we quantify how much faster this emulation approach is compared to other methods? Not without actually arriving at a similar result using other methods, and then repeating both procedures a number of times to get a statistically sound answer. This is not feasible to do because it would cost us a lot of additional computing time (and this would sort of defeat the purpose of this method, which was to avoid doing a large number of simulations). We can however make some rough estimates. The size of the interesting set according to the wave 3 emulator is about 0.75% of parameter space. Assuming that this emulator has correctly identified the part of parameter space most likely to contain (which we hope is a safe assumption since we tried to be very careful at every step), each simulation is now $\frac{1}{0.0075} \approx 133$ times more efficient than before. So as a lower limit, we have sped up exploration of parameter space by a factor 133.

Since the size of the interesting set, according to the wave 2 emulator is 1.66715%, we know that we can expect one in 60 randomly chosen inputs to be in this set. So we have 1 in 60 odds to find inputs which lead to a shoot dry weight of, on

average 19.63 g. Only 2 out of the 213 inputs in wave 3 that were distributed semi-randomly using a restricted Latin hypercube have a shoot dry weight above 23 g. This means that finding a root system architecture with a shoot dry weight of more than 23 g by trying random inputs would take $\frac{1}{0.0166715\frac{2}{213}} = 6388$ tries on average. When using 10 repetitions, this would mean 63880 42-day simulations. In contrast, we used about 7000 42-day simulations and 5000 20-day simulations. The 20-day simulations were on average more than 10 times faster so this is effectively 7500 42-day simulations, which resulted in 2 values above 23 g. This is a reduction of a factor $\frac{63880}{3750} = 17$, as compared to randomly searching. However, from the 5 points which the wave 2 emulator predicted would have the highest shoot dry weight, 4 had a shoot dry weight above 23 g and one was even above 24 g. These 5 points were selected by the wave 2 emulator as the best out of an initial set of 10 million points randomly placed around parameter space. Letting the emulator make predictions for this very large number of inputs took around 16 hours, about the same as a single OpenSimRoot simulation. Considering it would on average take 6388 randomly chosen inputs to expect one to have a shoot dry weight greater than 23 g, it is clear that this emulator-driven approach allows us to explore parameter space a lot faster than before.

Finally, the above apprimations are rough estimates of how much faster we can find inputs with a high shoot dry weight. But we do not just want to find the highest shoot dry weight, we want to find the region of parameter space where all the shoot dry weights within 5% of the maximum value reside. This is a much harder problem and any method to find this would require some sort of approximation of the model in question in order to map the relevant region of parameter space. For example, a genetic algorithm might find a path which leads to the top of a mountain, but getting a map of the mountaintop would still require extra work.

5.7 Biological interpretation

Our goal was to find root system architectures which led to high (the highest 5%) maize shoot dry weight in a challenging environment where plants are subjected to three nutrient stresses. Given the different conditions a phenotype should sat-

isfy in order to maximise uptake of different nutrients, we expected there to be a range of phenotypes which each lead to a high shoot dry weight through different combinations of phenes. So do we see such a variety in optimal phenotypes, are there any phenes for which certain values are always important to maximise the shoot dry weight and can we draw some conclusions about the optimal phenotypes?

Figure 5.22 shows representatives of the six best performing phenotypes, each of them having a mean shoot dry weight greater than 23 grams. While they do look similar in many respects, there are visual differences in axial root angle, nodal root number and branching densities.

Figure 5.23 shows the wave 3 shoot dry weights, projected onto each input dimension. For the seminal root number (SRN), nodal root number (NRN), seminal lateral branching frequency (SLBF), axial root angle (ARA) and major axes gravitropism (MAG), there is a clear preference for certain values. There are no phenotypes with low SRN in wave 3 at all, and a high value appears to be optimal. For NRN, there are no high values but the range of good values is a bit wider. For SLBF there is a clear peak in the middle and low and high values are both absent from wave 3. For ARA low values are absent, while for MAG high values do not appear.

The 5 phenes with a clear range of optimal values have been mentioned in the literature as important. Low axial (primary, seminal, nodal and brace) root number was hypothesized to be important in low nitrogen environments in [112], while increasing the axial root number increases the root length in the topsoil, which is good for phosphorus uptake. So it is not surprising that we see this in the importance of SRN and NRN. It is interesting however that they seem to have opposite optimal values, which is perhaps a consequence of the different pressures on them and the total number of nodal root numbers is in a sweet spot in the middle this way. Lateral branching frequency is a phene with opposite optimal values under nitrogen and phosphorus limitations [154] and seeing SLBF clearly prefer the middle could be a consequence of this being optimal under a combination of nitrogen and phosphorus deficiency. However, this raises the question why

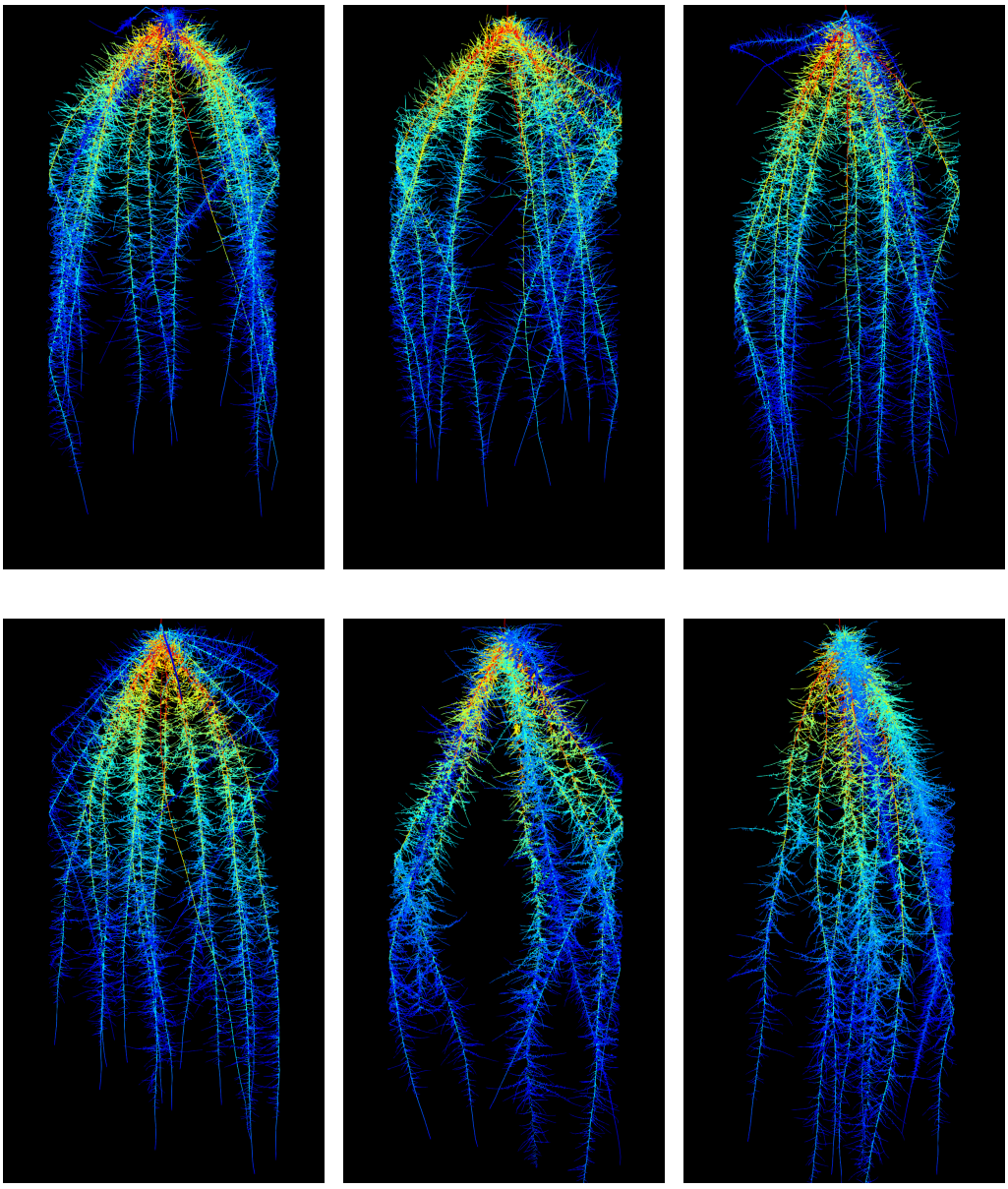


Figure 5.22: Representatives of the six maize phenotypes with the highest shoot dry weight at 42 days under nitrogen, phosphorus and potassium stress. The colours indicate root segment age with the youngest roots being blue and the oldest roots red. The views are approximately 150 cm from top to bottom.

none of the other branching frequencies (PLBF, NLBF, BLBF, CLBF, LLBF) show this same preference. The fact that three of these were marked as active variables means it is unlikely that the effects of these phenes on shoot dry weight is irrelevant and suggests there are different (combinations of) values which lead to a high shoot dry weight. Axial root angle has long been considered an important phene because it determines whether the root system mainly explores the topsoil, deeper soils or both [2, 41, 192]. The fact that we see high ARA values, which corresponds to shallow root angles, and low MAG values appear

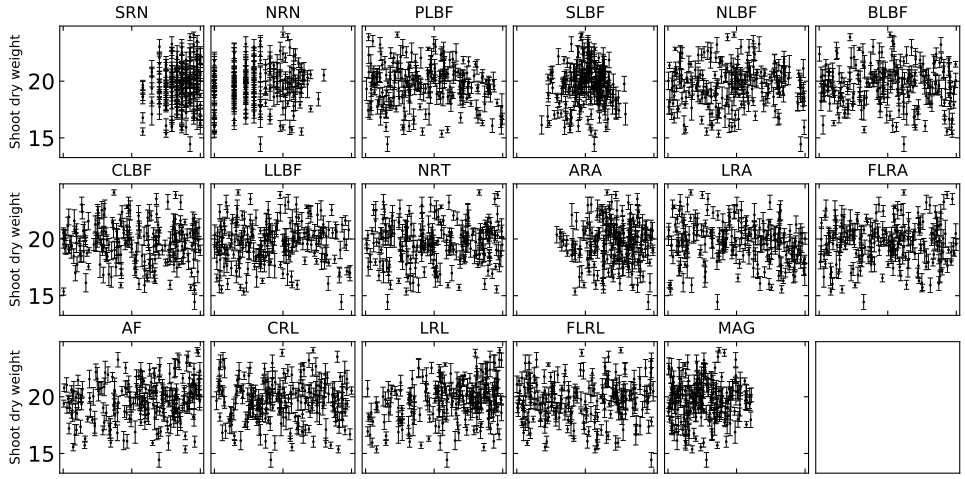
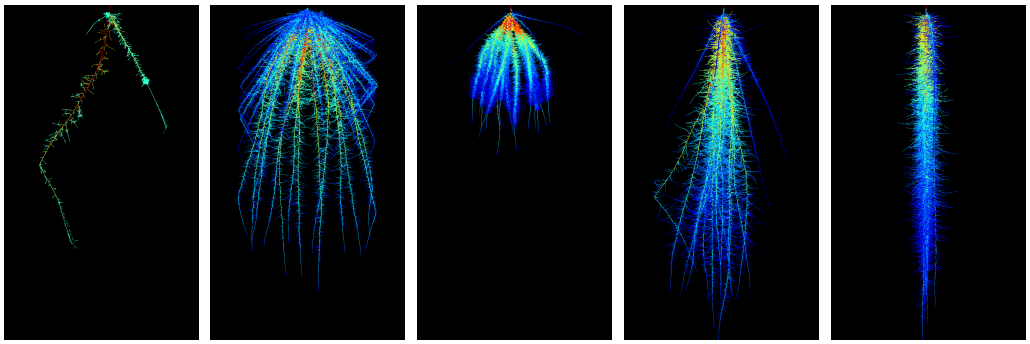


Figure 5.23: Mean shoot dry weights and standard deviations for wave 3 OpenSimRoot simulations, projected down onto each of the different inputs (phenes) under consideration.

as the best in the wave 3 simulations indicates that shallow and intermediate phenotypes are optimal and steep phenotypes are unlikely to correspond to high shoot dry weight. Initially all nutrients are concentrated in the topsoil and the immobility of phosphorus and, to a lesser extent, potassium mean that the largest concentrations will remain there. Nitrate does move down into deeper soil layers with water but the relationship we see implies that this does not create enough of an incentive for a steep root system.

Figure 5.24 shows what happens if we change any of the five phenes above to a value as far away from those of the best phenotype found so far. These 5 one-phene perturbations of the best phenotype found all performed significantly worse than the original phenotype, with shoot dry weights being 0.7 g, 19.2 g, 4.8 g, 13.4 g and 6.5 g respectively. They are also visually very distinct from any of the phenotypes in figure 5.22. The one-phene perturbation with higher nodal root number performed the best out of the 5, with a shoot dry weight of 19.2 g. This is likely due to the fact that the effect of this phene happens relatively late in development. If we could simulate for longer, we would likely see this phene have a greater effect on performance.

The importance of high SRN together with medium SLBF and low or medium NRN suggests that establishing a large root system early with a steady but lower increase in axial roots afterwards is a good strategy for maximising shoot dry



(a) The optimal phenotype but with lower seminal root number.

(b) The optimal phenotype but with higher nodal root number.

(c) The optimal phenotype but with higher seminal branching frequency.

(d) The optimal phenotype but with steeper axial root angle.

(e) The optimal phenotype but with a stronger gravitropic response.

Figure 5.24: Three representatives of one-phene perturbations of the best maize phenotype we found so far. They are identical to this best phenotype except for one phene. The colours indicate root segment age with the youngest roots being blue and the oldest roots red. The views are approximately 150 cm from top to bottom.

weight. Of course, since we are only simulating these root systems for 42 days, this means that we can only see the short term effects of some of the later nodal root whorls. Unfortunately the OpenSimRoot maize model has not yet been parametrised up to grain filling (or even flowering) so we have to base these conclusions on incomplete information. Still, it makes sense to start off with as large a root system as possible because this allows for greater nutrient extraction from the soil and the earlier a root emerges, the more nutrients it can extract per nutrient invested in the root. If carbon was limiting because the plants are also suffering from drought, this might not be the case because a large root system will require more carbon for maintenance. However, under just these nutrient limitations, establishing a big root system early seems to be a good strategy.

The 5 phenes which were not marked as active variables because they did not appear relevant when we optimised for BIC are brace root lateral branching frequency (BLBF), crown lateral (the laterals of brace roots) root branching frequency (CLBF), lateral root branching angle (LRA), fine lateral root branching angle (FLRA) and crown lateral root length (CRL). We see in Figure 5.23 that there is no obvious pattern indicating high or low values of these phenes are better for maximising shoot dry weight. The fact that these phenes have less explanatory power than the other phenes at this stage is somewhat expected. Since

brace roots appear relatively late in the simulation, one whorl emerges between day 20 and 30 and the other whorl between day 31 and 41, they have less time to affect plant development than for example the seminal roots, which appear together with the primary root. These roots also emerge in the topsoil, which other roots have had thorough time to explore by the time they emerge. Were we to simulate maize up to flowering, which happens approximately 70 days after germination, BLBF, CLBF and CRL would likely explain more of the variation. There is also a likely explanation why LRA and FLRA are not included as active inputs. While a 90 degree lateral root angle maximises the soil exploration volume and minimises competition between the parent root and the lateral, the effect of this on nutrient uptake is irrelevant when the roots in question are surrounded by many other roots of the same plant. This generally is the case for most laterals.

Out of the remaining phenes, primary lateral branching frequency (PLBF), nodal lateral branching frequency (NLBF), lateral root branching frequency (LLBF), nodal root timing (NRT), aerenchyma formation (AF), lateral root length (LRL) and fine lateral root length (FLRL), PLBF, AF and LRL appear to show a slight preference for either high or low values but we should be careful to draw any conclusions from what could well be an effect of our sampling procedure or random noise. Figures 5.25 and 5.26 show the two dimensional projections of the wave 3 emulator predictions. There are no obvious ridges or structures visible in these two dimensional projections. It is likely that the values of these 7 remaining phenes have significant impact on the shoot dry weight, because then they would not appear when optimising for BIC and the p-values associated with them would have been larger.

Perhaps continuing our analysis and running more waves of simulations will reveal more distinct structures in parameter space, but at this point the absence of a clear relationship between 7 of the phenes selected as active inputs and shoot dry weight implies that a plant is able to develop a large shoot when 3 or more of these phenes have synergistic values and that several different combinations exist (because otherwise they would show up in the one or two-dimensional projections). These combinations, or integrated phenotypes, could represent different local optima in the complex fitness landscape that evolution has optimised over.

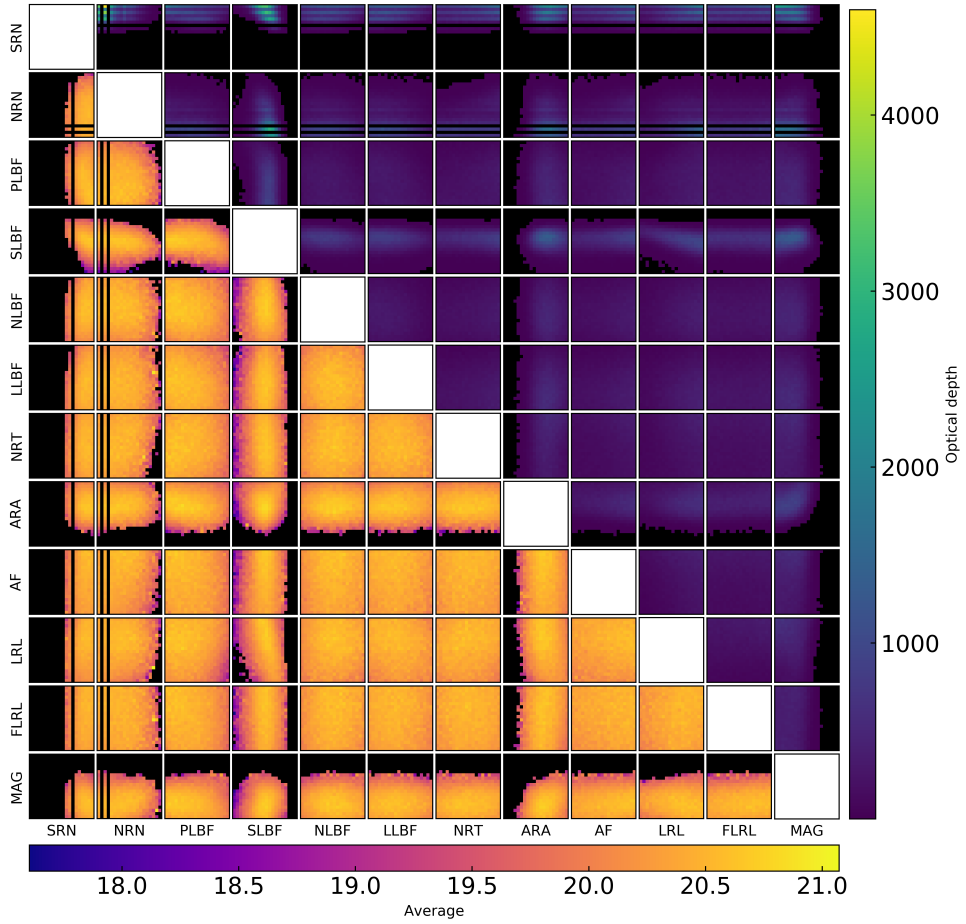


Figure 5.25: **An overview of the predictions of the wave 3 emulator of maize shoot dry weight after 42 days.** Each panel shows the projection onto two inputs (dimensions) of the average of emulator predictions for points in the interesting region, for panels below the diagonal, or the optical depth, which is the number of points in the interesting region, for panels above the diagonal, with each pixel representing a number of points for which the values in the two dimensions in question match the x, y coordinates in the panel in question. Black regions contain no points in the interesting region (or, in the case of the first two inputs, which are discrete, are not part of parameter space). Only the 12 active inputs are displayed.

As a simple example, consider the following: Root surface area in the topsoil is important for phosphorus uptake, so we expect all the phenotypes in wave 3 to have a reasonably large amount of roots in the top soil. But this can be achieved through many different combinations of phenes. For example, the amount of lateral root surface area subtending from seminal roots is proportional to the number of seminal roots, multiplied by the average lateral root branching density, multiplied by the average lateral root length. Each of these can be varied independently to obtain the same result. And we have multiple classes of axial roots (primary, seminal, nodal, brace), each with different branching densities,

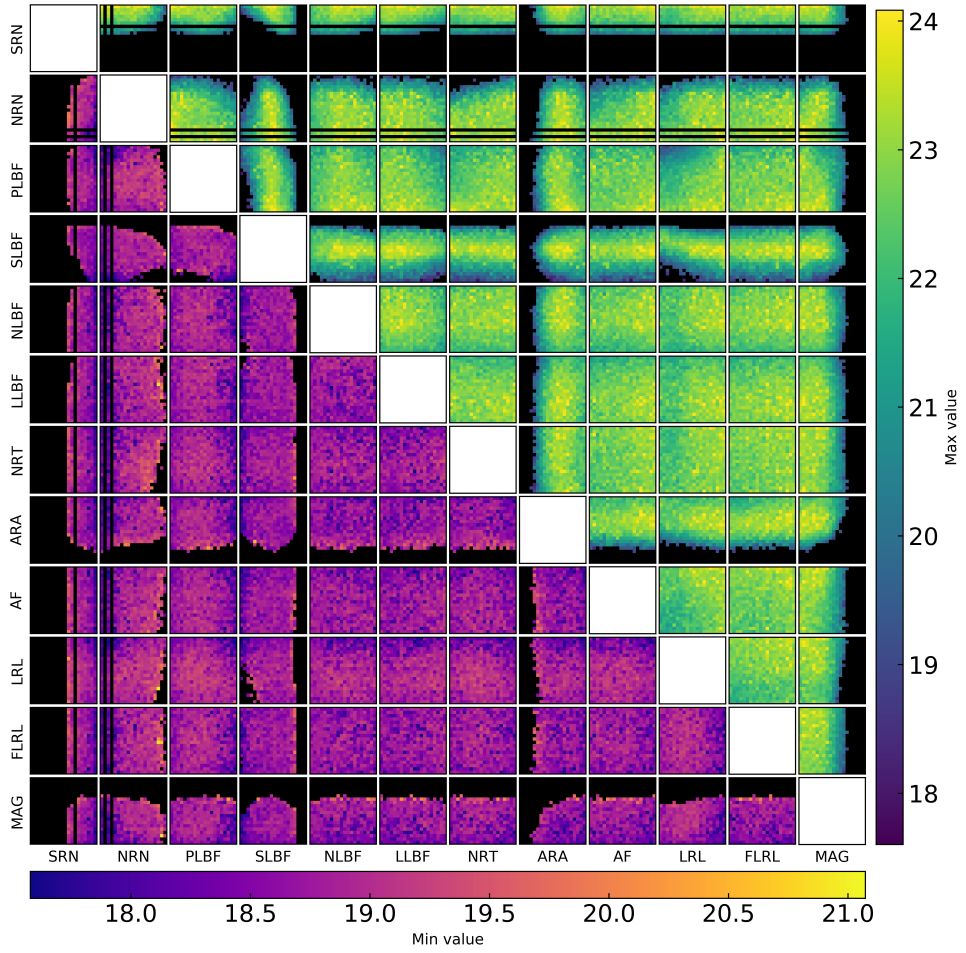


Figure 5.26: **An overview of the predictions of the wave 3 emulator of maize shoot dry weight after 42 days.** Each panel shows the projection onto two inputs (dimensions) of the minimum of points in the interesting region (for panels below the diagonal) or the maximum of points in the interesting region (above the diagonal), with each pixel representing a number of points for which the values in the two dimensions in question match the x, y coordinates in the panel in question. Black regions contain no points in the interesting region (or, in the case of the first two inputs, which are discrete, are not part of parameter space). Only the 12 active inputs are displayed.

from the laterals even smaller fine lateral roots emerge and the lengths of laterals and fine laterals can vary. This gives us many different phene combinations to obtain the same lateral root surface area.

If we were maximising lateral root surface area this would make it easy to figure out the optimal phene combinations, but since roots require nutrients and carbon to grow and maintain, it is not so simple. A detailed statistical analysis of root system outputs at 42 days will perhaps reveal if there are quantities, calculable from phene values, which explain why certain integrated phenotypes lead to high

shoot dry weight (or minimise/maximise some other output). Since this is the holy grail of root system architecture and many researchers have spent entire careers chipping away at this question we do not expect a definitive answer, but undoubtedly this would increase our understanding of the relationships between plant development and root system architectural traits.

Chapter 6

Conclusion

OpenSimRoot is a collaborative and open source effort to produce a feature-rich plant model with a focus on roots. We actively developed new capabilities in order to capture more features of reality and increase the scope of questions that can be addressed with it. In addition to the research we conducted ourselves, we hope that these new capabilities will give future researchers more tools to address their research questions.

The research described in this thesis lead to some collateral outputs which did not fit into any of the chapters. While learning about OpenSimRoot, a guide, included in the appendix, was written in order to help future OpenSimRoot developers and users get started. It includes a description of important models, how the engine works, information about the application programming interface (API) and other helpful information. A workshop was given to a small number of root researchers while on a visit to Canberra, Australia, where participants managed to get some basic models of species of interest up and running over the course of a few days. A graphical user interface (GUI) was developed to make creating OpenSimRoot input files easier for new users, as well as a number of tools for quickly editing input files or generating the files for large simulation experiments. These have been made public on GitLab [1]. Finally, while working on OpenSimRoot, a number of improvements were made to the code, in collaboration with the other OpenSimRoot developers. One particularly important example was that an update had the unforeseen effect of altering root angle changes due to gravitropism. This was addressed and an automated test was added to make sure

[1] https://gitlab.com/rootmodels/opensimroottools

that a similar situation will automatically be flagged in the future.

We implemented a root loss module so that OpenSimRoot can simulate root loss going forward. With this new functionality we simulated a number of barley, bean and maize phenotypes in soils with differing nutrient availabilities subjected to different types and levels of root loss. Plants which were phosphorus stressed had their development impaired significantly by root loss, where the effect on nitrogen-stressed plants was less detrimental and for some maize phenotypes even slightly beneficial if only lateral roots were subjected to root loss. Rather than the absolute amount of root length lost, the timing and localisation of root loss was important for determining the effect on plant development. It was especially detrimental for plants to lose a large fraction of their total roots early during development. Phenotypes with a high number of axial roots 10 days after germination were more resilient to axial root loss. This might also explain why maize and bean were more resilient to axial root loss than barley. Finally, it was interesting to see that the optimal root phenotypes without root loss were generally also the optimal phenotypes with root loss.

We also implemented more sophisticated models for C3 and C4 photosynthesis and stomatal responses to drought. Where previously drought had no effect on plant development in OpenSimRoot, now it will lead to a lower stomatal conductance, lower carbon dioxide concentrations in the leaves and lower photosynthesis rates. The implementation of the relevant models required some modifications to the OpenSimRoot engine and the addition of a numerical root finder. We used this new functionality to compare a reference phenotype with a more parsimonious phenotype under well watered and drought conditions. The parsimonious phenotype took up slightly less water but managed to grow a bigger shoot under drought conditions. It achieved this through higher carbon efficiency, that is, for every gram of carbon spent on roots, it took up more water from the soil than the reference phenotype. It is only through modelling that we can keep track of the carbon budget of plants over time and pin down exactly why one root system outperforms another.

In the final chapter we used an emulator constructed with statistical machine

learning techniques to maximise the shoot dry weight of maize plant system in a low-nitrogent, low-phosphorus, low-potassium soil over 17 different root properties. By running successive sets of simulations, 'waves', training a new emulator for each of these and then restricting our attention to specific parts of parameter space based on the predictions of the emulator we slowly zero in on the optimal root systems. This technique has been used to find model parameters that reproduce experimental data in a number of different contexts, including biological but not for trait optimisation. We iterated on the technique by using the fact that OpenSimRoot produces outputs at regular intervals, which are highly correlated. This allowed us to use outputs at 20 days to predict outputs at 42 days and further reduce the computational resources needed to make predictions. With this emulator-driven approach, we found root system architectures which lead to shoot dry weights under nutrient stress only about 20% less than the reference phenotype achieved in soil with abundant nutrient availability. This was all done with about 7000 full length simulations and 5000 20-day simulations, which includes 10 repetitions for each input. Considering the size of a 17 dimensional space, the relatively small number of simulations shows the potential of this approach in biological contexts. For comparison, to even get a single root system phenotype with a shoot dry weight within 5% of the best performer we found, we estimate one would need around 63880 full length simulations on average (including 10 repetitions) if one was searching parameter space randomly. This does not even take into account the fact that the final emulator lets us generate potentially good phenotypes with relative ease. The final results confirmed that axial root number and angle and lateral branching density are important traits while also implying that there are many different integrated phenotypes among the best performers.

Bringing this all together, there are many connections that can be drawn between the different subjects in this thesis, leading to new research questions. Drought can increase root mortality and in turn root loss decreases the ability of plants to access soil water. Perhaps the parsimonious phenotype we found to perform better under drought is more vulnerable to root loss than a more expansive root system and the optimal phenotype depends on the exact balance between these stresses. Adding water requirements for plants to grow and maintain homeostasis

would allow us to look at interesting questions concerning root loss because this introduces new trade offs. Roots would not only require carbon but also water for maintenance and growth and it would be interesting to see in what cases it becomes better to lose roots and regrow them versus actively maintaining them.

We only studied a limited number of different root systems, varying only 3 properties in the root loss chapter. Because of the factorial design and the different environments this already represented a large number of simulations. In the drought chapter we considered only 2 different root systems. Using the emulator approach, we could search for optimal root system architectures in root loss and drought scenarios more efficiently. We could also find combinations of root traits which maximise our objectives that we would never think of otherwise. The emulation method can also be useful in parametrising OpenSimRoot models, since it has already been used in other contexts to match models to data. The drought module will be used together with a newly developed soil impedance module in order to study plant development in the context of hardening soils as well as a number of other simulation studies.

Our research highlights the potential of computational approaches in tackling the complexity of root systems and the importance of continued research into this hidden half of plants, especially in the context of a changing climate and the agricultural challenges ahead of us.

Bibliography

[1] I. J. Alexander and R. I. Fairley. "Effects of N fertilisation on populations of fine roots and mycorrhizas in spruce humus". In: *Plant and Soil* 71.1-3 (Feb. 1983), pp. 49–53. DOI: 10.1007/bf02182640.

[2] M. L. Ali et al. "Genetic variation in seminal and nodal root angle and their association with grain yield of maize under water-stressed field conditions". In: *Plant and Soil* 397.1-2 (Aug. 2015), pp. 213–225. DOI: 10.1007/s11104-015-2554-x.

[3] R. G. Allen et al. "Crop evapotranspiration-Guidelines for computing crop water requirements-FAO Irrigation and drainage paper 56". In: *FAO, Rome* 300.9 (1998), p. D05109.

[4] D. M. Alm, J. Cavelier, and P. S. Nobel. "A Finite-element Model of Radial and Axial Conductivities for Individual Roots: Development and Validation for Two Desert Succulents". In: *Annals of Botany* 69.1 (Jan. 1992), pp. 87–92. DOI: 10.1093/oxfordjournals.aob.a088311.

[5] B. Amos and D. Walters. "Maize root biomass and net rhizodeposited carbon". In: *Soil Science Society of America Journal* 70.5 (2006), pp. 1489–1503. DOI: 10.2136/sssaj2005.0216.

[6] I. Andrianakis et al. "Bayesian history matching of complex infectious disease models using emulation: a tutorial and a case study on HIV in Uganda". In: *PLoS Comput Biol* 11.1 (2015), e1003968. DOI: 10.1371/journal.pcbi.1003968.

[7] Y. Assefa et al. "Analysis of Long Term Study Indicates Both Agronomic Optimal Plant Density and Increase Maize Yield per Plant Contributed to Yield Gain". In: *Scientific Reports* 8.1 (Mar. 2018). DOI: 10.1038/s41598-018-23362-x.

[8] O. K. Atkin, B. Botman, and H. Lambers. “The Causes of Inherently Slow Growth in Alpine Plants: An Analysis Based on the Underlying Carbon Economies of Alpine and Lowland Poa Species”. In: *Functional Ecology* 10.6 (Dec. 1996), p. 698. DOI: 10.2307/2390504.

[9] J. A. Atkinson et al. “Branching Out in Roots: Uncovering Form, Function, and Regulation”. In: *Plant Physiology* 166.2 (Aug. 2014), pp. 538–550. DOI: 10.1104/pp.114.245423.

[10] U. Baetz and E. Martinoia. “Root exudates: the hidden part of plant defense”. In: *Trends in Plant Science* 19.2 (Feb. 2014), pp. 90–98. DOI: 10.1016/j.tplants.2013.11.006.

[11] P. A. H. M. Bakker et al. “The rhizosphere revisited: root microbiomics”. In: *Frontiers in Plant Science* 4 (2013). DOI: 10.3389/fpls.2013.00165.

[12] A. V. Barker and D. J. Pilbeam. *Handbook of plant nutrition*. CRC press, 2015.

[13] J. Benjamin et al. “Water deficit stress effects on corn (Zea mays, L.) root: shoot ratio”. In: *Open Journal of Soil Science* 2014 (2014). DOI: 10.4236/ojss.2014.44018.

[14] R. E. Blankenship. *Molecular mechanisms of photosynthesis*. John Wiley & Sons, 2014.

[15] S. Blossfeld et al. “The dynamics of oxygen concentration, pH value, and organic acids in the rhizosphere of Juncus spp.” In: *Soil Biology and Biochemistry* 43.6 (June 2011), pp. 1186–1197. DOI: 10.1016/j.soilbio.2011.02.007.

[16] A. Blum. “Crop responses to drought and the interpretation of adaptation”. In: *Drought Tolerance in Higher Plants: Genetical, Physiological and Molecular Biological Analysis*. Springer Netherlands, 1996, pp. 57–70. DOI: 10.1007/978-94-017-1299-6_8.

[17] W. Böhm. *Methods of studying root systems*. Vol. 33. Springer Science & Business Media, 2012.

[18] K. D. Bonifas et al. “Nitrogen supply affects root: shoot ratio in corn and velvetleaf (Abutilon theophrasti)”. In: *Weed Science* 53.5 (2005), pp. 670–675. DOI: 10.1614/WS-05-002R.1.

[19] A. Bonneu et al. “A minimal continuous model for simulating growth and development of plant root systems”. In: *Plant and Soil* 354.1-2 (Dec. 2011), pp. 211–227. DOI: 10.1007/s11104-011-1057-7.

[20] A. Boukouvalas et al. “Bayesian Precalibration of a Large Stochastic Microsimulation Model”. In: *IEEE Transactions on Intelligent Transportation Systems* 15.3 (June 2014), pp. 1337–1347. DOI: 10.1109/tits.2014.2304394.

[21] T. J. Bouma et al. “Estimating age-dependent costs and benefits of roots with contrasting life span: comparing apples and oranges”. In: *New Phytologist* 150.3 (June 2001), pp. 685–695. DOI: 10.1046/j.1469-8137.2001.00128.x.

[22] G. Bowes. “Growth at elevated CO2: photosynthetic responses mediated through Rubisco”. In: *Plant, Cell and Environment* 14.8 (Oct. 1991), pp. 795–806. DOI: 10.1111/j.1365-3040.1991.tb01443.x.

[23] M. C. Brundrett. “Coevolution of roots and mycorrhizas of land plants”. In: *New phytologist* 154.2 (2002), pp. 275–304. DOI: 10.1046/j.1469-8137.2002.00397.x.

[24] D. R. Bryla, T. J. Bouma, and D. M. Eissenstat. “Root respiration in citrus acclimates to temperature and slows during drought”. In: *Plant, Cell and Environment* 20.11 (Nov. 1997), pp. 1411–1420. DOI: 10.1046/j.1365-3040.1997.d01-36.x.

[25] T. N. Buckley. “How do stomata respond to water status?” In: *New Phytologist* 224.1 (2019), pp. 21–36. DOI: 10.1111/nph.15899.

[26] A. Burton, K. Pregitzer, and R. Hendrick. “Relationships between fine root dynamics and nitrogen availability in Michigan northern hardwood forests”. In: *Oecologia* 125.3 (Nov. 2000), pp. 389–399. DOI: 10.1007/s004420000455.

[27] A. L. Burton, K. M. Brown, and J. P. Lynch. “Phenotypic diversity of root anatomical and architectural traits in Zea species”. In: *Crop Science* 53.3 (2013), pp. 1042–1055. DOI: 10.2135/cropsci2012.07.0440.

[28] K. Butterbach-Bahl, H. Papen, and H. Rennenberg. “Impact of gas transport through rice cultivars on methane emission from rice paddy fields”.

In: *Plant, Cell and Environment* 20.9 (Sept. 1997), pp. 1175–1183. DOI: 10.1046/j.1365-3040.1997.d01-142.x.

[29] H. Cai et al. "Mapping QTLs for root system architecture of maize (Zea mays L.) in the field at different developmental stages". In: *Theoretical and Applied Genetics* 125.6 (2012), pp. 1313–1324. DOI: 10.1007/s00122-012-1915-6.

[30] M. M. Caldwell and J. H. Richards. "Hydraulic lift: water efflux from upper roots improves effectiveness of water uptake by deep roots". In: *Oecologia* 79.1 (Apr. 1989), pp. 1–5. DOI: 10.1007/bf00378231.

[31] P. Castro, J. Puertolas, and I. C. Dodd. "Stem girdling uncouples soybean stomatal conductance from leaf water potential by enhancing leaf xylem ABA concentration". In: *Environmental and Experimental Botany* 159 (Mar. 2019), pp. 149–156. DOI: 10.1016/j.envexpbot.2018.12.020.

[32] F. S. Chapin, R. H. Groves, and L. T. Evans. "Physiological determinants of growth rate in response to phosphorus supply in wild and cultivated Hordeum species". In: *Oecologia* 79.1 (Apr. 1989), pp. 96–105. DOI: 10.1007/bf00378245.

[33] Y. L. Chen et al. "Modelling root plasticity and response of narrow-leafed lupin to heterogeneous phosphorus supply". In: *Plant and Soil* 372.1-2 (May 2013), pp. 319–337. DOI: 10.1007/s11104-013-1741-x.

[34] Y. L. Chen et al. "Phenotypic variability and modelling of root structure of wild Lupinus angustifolius genotypes". In: *Plant and Soil* 348.1-2 (2011), p. 345. DOI: 10.1007/s11104-011-0939-z.

[35] W. Cheng, D. C. Coleman, and J. E. Box. "Measuring root turnover using the minirhizotron technique". In: *Agriculture, Ecosystems & Environment* 34.1-4 (Feb. 1991), pp. 261–267. DOI: 10.1016/0167-8809(91)90113-c.

[36] J. G. Chimungu et al. "Utility of root cortical aerenchyma under water limited conditions in tropical maize (Zea mays L.)" In: *Field Crops Research* 171 (Feb. 2015), pp. 86–98. DOI: 10.1016/j.fcr.2014.10.009.

[37] A. Christmann et al. "A hydraulic signal in root-to-shoot signalling of water shortage". In: *The Plant Journal* 52.1 (July 2007), pp. 167–174. DOI: 10.1111/j.1365-313x.2007.03234.x.

[38] V. Clausnitzer and J. W. Hopmans. "Simultaneous modeling of transient three-dimensional root growth and soil water flow". In: *Plant and Soil* 164.2 (July 1994), pp. 299–314. DOI: 10.1007/bf00010082.

[39] A. B. Cousins et al. "Simultaneous determination of Rubisco carboxylase and oxygenase kinetic parameters in Triticum aestivum and Zea mays using membrane inlet mass spectrometry". In: *Plant, Cell & Environment* 33.3 (2010), pp. 444–452. DOI: 10.1111/j.1365-3040.2009.02095.x.

[40] P. S. Craig et al. "Pressure matching for hydrocarbon reservoirs: a case study in the use of Bayes linear strategies for large computer experiments". In: *Case studies in Bayesian statistics*. Springer, 1997, pp. 37–93. DOI: 10.1007/978-1-4612-2290-3_2.

[41] A. Dathe et al. "Impact of axial root growth angles on nitrogen acquisition in maize depends on environmental conditions". In: *Annals of botany* 118.3 (2016), pp. 401–414. DOI: 10.1093/aob/mcw112.

[42] J. D. Deans and E. D. Ford. "Modelling root structure and stability". In: *Tree Root Systems and Their Mycorrhizas*. Springer Netherlands, 1983, pp. 189–195. DOI: 10.1007/978-94-009-6833-2_19.

[43] A. Diggle. "ROOTMAP—a model in three-dimensional coordinates of the growth and structure of fibrous root systems". In: *Plant and soil* 105.2 (1988), pp. 169–178. DOI: 10.1007/BF02376780.

[44] K. Dong et al. "Scalable Log Determinants for Gaussian Process Kernel Learning". In: *Advances in Neural Information Processing Systems 30 (NIPS), 2017* (Nov. 9, 2017). arXiv: 1711.03481v1 [stat.ML].

[45] C. Doussan. "Modelling of the Hydraulic Architecture of Root Systems: An Integrated Approach to Water Absorption—Model Description". In: *Annals of Botany* 81.2 (Feb. 1998), pp. 213–223. DOI: 10.1006/anbo.1997.0540.

[46] V. M. Dunbabin et al. "Identifying fertiliser management strategies to maximise nitrogen and phosphorus acquisition by wheat in two contrasting soils from Victoria, Australia". In: *Soil Research* 47.1 (2009), p. 74. DOI: 10.1071/sr08107.

[47] V. M. Dunbabin et al. “Modelling the interactions between water and nutrient uptake and root growth”. In: *Plant and Soil* 239.1 (2002), pp. 19–38. DOI: 10.1023/A:1014939512104.

[48] V. M. Dunbabin et al. “Modelling root-soil interactions using three-dimensional models of root growth, architecture and function”. In: *Plant and Soil* 372.1-2 (June 2013), pp. 93–124. DOI: 10.1007/s11104-013-1769-y.

[49] W. G. Duncan and J. D. Hesketh. “Net Photosynthetic Rates, Relative Leaf Growth Rates, and Leaf Numbers of 22 Races of Maize Grown at Eight Temperatures 1”. In: *Crop Science* 8.6 (Nov. 1968), pp. 670–674. DOI: 10.2135/cropsci1968.0011183x000800060009x.

[50] L. X. Dupuy et al. “A generic 3D finite element model of tree anchorage integrating soil mechanics and real root system architecture”. In: *American Journal of Botany* 94.9 (Sept. 2007), pp. 1506–1514. DOI: 10.3732/ajb.94.9.1506.

[51] L. Dupuy, P. J. Gregory, and A. G. Bengough. “Root growth models: towards a new generation of continuous approaches”. In: *Journal of Experimental Botany* 61.8 (Jan. 2010), pp. 2131–2143. DOI: 10.1093/jxb/erp389.

[52] D. Eapen et al. “Hydrotropism: root growth responses to water”. In: *Trends in plant science* 10.1 (2005), pp. 44–50. DOI: 10.1016/j.tplants.2004.11.004.

[53] S. Egli and I. Kälin. “19 Root Window Technique for in vivo Observation of Ectomycorrhiza on Forest Trees”. In: *Methods in Microbiology*. Elsevier BV, 1991, pp. 423–433. DOI: 10.1016/s0580-9517(08)70189-2.

[54] D. M. Eissenstat et al. “Recovery of citrus surface roots following prolonged exposure to dry soil”. In: *Journal of Experimental Botany* 50.341 (Dec. 1999), pp. 1845–1854. DOI: 10.1093/jxb/50.341.1845.

[55] D. Eissenstat and R. Yanai. “The Ecology of Root Lifespan”. In: *Advances in Ecological Research Volume 27*. Elsevier BV, 1997, pp. 1–60. DOI: 10.1016/s0065-2504(08)60005-7.

[56] J. F. Espeleta, D. M. Eissenstat, and J. H. Graham. “Citrus root responses to localized drying soil: A new approach to studying mycorrhizal effects

on the roots of mature trees". In: *Plant and Soil* 206.1 (1998), pp. 1–10. DOI: 10.1023/a:1004325300583.

[57] D. E. Evans. "Aerenchyma formation". In: *New Phytologist* 161.1 (Oct. 2003), pp. 35–49. DOI: 10.1046/j.1469-8137.2003.00907.x.

[58] G. D. Farquhar, S. v. von Caemmerer, and J. A. Berry. "A biochemical model of photosynthetic CO 2 assimilation in leaves of C 3 species". In: *planta* 149.1 (1980), pp. 78–90. DOI: 10.1007/BF00386231.

[59] G. D. Farquhar and S. Von Caemmerer. "Modelling of photosynthetic response to environmental conditions". In: *Physiological plant ecology II*. Springer, 1982, pp. 549–587. DOI: 10.1007/978-3-642-68150-9_17.

[60] A. Fick. "V. On liquid diffusion". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 10.63 (1855), pp. 30–39. DOI: 10.1080/14786445508641925.

[61] M. C. T. Fisher, D. M. Eissenstat, and J. P. Lynch. "Lack of evidence for programmed root senescence in common bean (Phaseolus vulgaris) grown at different levels of phosphorus supply". In: *New Phytologist* 153.1 (Jan. 2002), pp. 63–71. DOI: 10.1046/j.0028-646x.2001.00285.x.

[62] A. H. Fitter et al. "Architectural analysis of plant root systems 1. Architectural correlates of exploitation efficiency". In: *New Phytologist* 118.3 (July 1991), pp. 375–382. DOI: 10.1111/j.1469-8137.1991.tb00018.x.

[63] P. Forbes, K. Black, and J. Hooker. "Temperature-induced alteration to root longevity in Lolium perenne". In: *Plant and Soil* 190.1 (1997), pp. 87–90. DOI: 10.1023/a:1004298804353.

[64] A. L. Fredeen, I. M. Rao, and N. Terry. "Influence of Phosphorus Nutrition on Growth and Carbon Partitioning in Glycine max". In: *Plant Physiology* 89.1 (Jan. 1989), pp. 225–230. DOI: 10.1104/pp.89.1.225.

[65] W. J. Gale and C. A. Cambardella. "Carbon Dynamics of Surface Residue- and Root-derived Organic Matter under Simulated No-till". In: *Soil Science Society of America Journal* 64.1 (Jan. 2000), pp. 190–195. DOI: 10.2136/sssaj2000.641190x.

[66] T. Galindo-Castañeda, K. M. Brown, and J. P. Lynch. "Reduced root cortical burden improves growth and grain yield under low phosphorus availability in maize". In: *Plant, Cell & Environment* 41.7 (May 2018), pp. 1579–1592. DOI: 10.1111/pce.13197.

[67] A. C. Gange, V. K. Brown, and G. S. Sinclair. "Reduction of black vine weevil larval growth by vesicular-arbuscular mycorrhizal infection". In: *Entomologia Experimentalis et Applicata* 70.2 (Feb. 1994), pp. 115–119. DOI: 10.1111/j.1570-7458.1994.tb00739.x.

[68] Z. Ge, G. Rubio, and J. P. Lynch. "The importance of root gravitropism for inter-root competition and phosphorus acquisition efficiency: results from a geometric simulation model". In: *Plant and Soil* 218.1-2 (2000), pp. 159–171. DOI: 10.1023/A:1014987710937.

[69] M. T. van Genuchten. "A Closed-form Equation for Predicting the Hydraulic Conductivity of Unsaturated Soils". In: *Soil Science Society of America Journal* 44.5 (Sept. 1980), pp. 892–898. DOI: 10.2136/sssaj1980.03615995004400050002x.

[70] A. Gerwitz and E. R. Page. "An Empirical Mathematical Model to Describe Plant Root Systems". In: *The Journal of Applied Ecology* 11.2 (Aug. 1974), p. 773. DOI: 10.2307/2402227.

[71] R. A. Gill and R. B. Jackson. "Global patterns of root turnover for terrestrial ecosystems". In: *New Phytologist* 147.1 (July 2000), pp. 13–31. DOI: 10.1046/j.1469-8137.2000.00681.x.

[72] C. Godin and H. Sinoquet. "Functional-structural plant modelling". In: *New Phytologist* 166.3 (May 2005), pp. 705–708. DOI: 10.1111/j.1469-8137.2005.01445.x.

[73] T. Gollan et al. "Soil Water Status Affects the Stomata1". In: *Functional Plant Biology* 13.4 (1986), p. 459. DOI: 10.1071/pp9860459.

[74] D. Gonzalez, J. Postma, and M. Wissuwa. "Cost-Benefit Analysis of the Upland-Rice Root Architecture in Relation to Phosphate: 3D Simulations Highlight the Importance of S-Type Lateral Roots for Reducing the Pay-Off Time". In: *Frontiers in Plant Science* 12 (Mar. 2021). DOI: 10.3389/fpls.2021.641835.

[75] D. Hayes and T. Seastedt. “Root dynamics of tallgrass prairie in wet and dry years”. In: *Botany* 65.4 (1987), pp. 787–791. DOI: 10.1139/b87-105.

[76] W. M. Haynes. *CRC handbook of chemistry and physics*. CRC press, 2014.

[77] R. L. Hendrick and K. S. Pregitzer. “Patterns of fine root mortality in two sugar maple forests”. In: *Nature* 361 (1993), p. 7. DOI: 10.1038/361059a0.

[78] M. Henke et al. “Exploring root developmental plasticity to nitrogen with a three-dimensional architectural model”. In: *Plant and Soil* 385.1-2 (Aug. 2014), pp. 49–62. DOI: 10.1007/s11104-014-2221-7.

[79] D. Hillel. *Fundamentals of Soil Physics*. Elsevier, 1980. DOI: 10.1016/c2009-0-03109-2.

[80] M. D. Ho et al. “Root architectural tradeoffs for water and phosphorus acquisition”. In: *Functional Plant Biology* 32.8 (2005), p. 737. DOI: 10.1071/fp05043.

[81] J. E. Hobbie and E. A. Hobbie. “15N in Symbiotic Fungi and Plants Estimates Nitrogen and Carbon Flux Rates in Arctic Tundra”. In: *Ecology* 87.4 (Apr. 2006), pp. 816–822. DOI: 10.1890/0012-9658(2006)87[816:nisfap]2.0.co;2.

[82] N. Hoecker et al. “Manifestation of heterosis during early maize (Zea mays L.) root development”. In: *Theoretical and Applied Genetics* 112.3 (2006), pp. 421–429. DOI: 10.1007/s00122-005-0139-4.

[83] D. P. Holzworth et al. “APSIM – Evolution towards a new generation of agricultural systems simulation”. In: *Environmental Modelling & Software* 62 (Dec. 2014), pp. 327–350. DOI: 10.1016/j.envsoft.2014.07.009.

[84] B. Hu et al. “Root cortical aerenchyma inhibits radial nutrient transport in maize (Zea mays)”. In: *Annals of Botany* 113.1 (Nov. 2013), pp. 181–189. DOI: 10.1093/aob/mct259.

[85] A. Hund et al. “Root morphology and photosynthetic performance of maize inbred lines at low temperature”. In: *European Journal of Agronomy* 27.1 (2007), pp. 52–61. DOI: 10.1016/j.eja.2007.01.003.

[86] K. Ikegami et al. “Activation of abscisic acid biosynthesis in the leaves of Arabidopsis thaliana in response to water deficit”. In: *Journal of Plant Research* 122.2 (Dec. 2008), pp. 235–243. DOI: 10.1007/s10265-008-0201-9.

[87] P. Ineson et al. “Root production and turnover in an upland grassland subjected to artificial soil warming respond to radiation flux and nutrients, not temperature”. In: *Oecologia* 120.4 (Sept. 1999), pp. 575–581. DOI: 10.1007/s004420050892.

[88] K. Ito et al. “Lateral root development, including responses to soil drying, of maize (Zea mays) and wheat (Triticum aestivum) seminal roots”. In: *Physiologia Plantarum* 127.2 (2006), pp. 260–267. DOI: 10.1111/j.1399-3054.2006.00657.x.

[89] S. Itoh and S. A. Barber. “A numerical solution of whole plant nutrient uptake for soil-root systems with root hairs”. In: *Plant and Soil* 70.3 (Oct. 1983), pp. 403–413. DOI: 10.1007/bf02374895.

[90] R. B. Jackson and M. M. Caldwell. “The Scale of Nutrient Heterogeneity Around Individual Plants and Its Quantification with Geostatistics”. In: *Ecology* 74.2 (Mar. 1993), pp. 612–614. DOI: 10.2307/1939320.

[91] M. Javaux et al. “Use of a Three-Dimensional Detailed Modeling Approach for Predicting Root Water Uptake”. In: *Vadose Zone Journal* 7.3 (2008), p. 1079. DOI: 10.2136/vzj2007.0115.

[92] X. Jia, P. Liu, and J. P. Lynch. “Greater lateral root branching density in maize improves phosphorus acquisition from low phosphorus soil”. In: *Journal of Experimental Botany* 69.20 (July 2018), pp. 4961–4970. DOI: 10.1093/jxb/ery252.

[93] G. Karssen, W. Wesemael, and M. Moens. “Root-knot nematodes.” In: *Plant nematology*. CABI, 2013, pp. 73–108. DOI: 10.1079/9781780641515.0073.

[94] D. B. Kell. “Large-scale sequestration of atmospheric carbon via plant roots in natural and agricultural ecosystems: why and how”. In: *Phil. Trans. R. Soc. B* 367.1595 (2012), pp. 1589–1597. DOI: 10.1098/rstb.2011.0244.

[95] A. A. M. Khalil and J. Grace. “Does Xylem Sap ABA Control the Stomatal Behaviour of Water-Stressed Sycamore (Acer pseudoplatanusL.) Seedlings?” In: *Journal of Experimental Botany* 44.7 (1993), pp. 1127–1134. DOI: 10.1093/jxb/44.7.1127.

[96] J. S. King, K. S. Pregitzer, and D. R. Zak. “Clonal variation in above- and below-ground growth responses of Populus tremuloides Michaux: Influence of soil warming and nutrient availability”. In: *Plant and Soil* 217.1/2 (1999), pp. 119–130. DOI: 10.1023/a:1004560311563.

[97] P. E. Kriedemann et al. “Abscisic Acid and Stomatal Regulation”. In: *Plant Physiology* 49.5 (May 1972), pp. 842–847. DOI: 10.1104/pp.49.5.842.

[98] U. Kutschera, R. Pieruschka, and J. A. Berry. “Leaf development, gas exchange characteristics, and photorespiratory activity in maize seedlings”. In: *Photosynthetica* 48.4 (Dec. 2010), pp. 617–622. DOI: 10.1007/s11099-010-0079-3.

[99] Y. Kuzyakov and G. Domanski. “Carbon input by plants into the soil. Review”. In: *Journal of Plant Nutrition and Soil Science* 163.4 (2000), pp. 421–431. ISSN: 1522-2624. DOI: 10.1002/1522-2624(200008)163:4<421::AID-JPLN421>3.0.CO;2-R.

[100] F. Laio, P. D’Odorico, and L. Ridolfi. “An analytical model to relate the vertical root distribution to climate and soil properties”. In: *Geophysical Research Letters* 33.18 (Sept. 2006), n/a–n/a. DOI: 10.1029/2006gl027331.

[101] H. Lambers and R. S. Oliveira. *Plant Physiological Ecology*. Springer International Publishing, 2019. DOI: 10.1007/978-3-030-29639-1.

[102] D. Leitner et al. “A dynamic root system growth model based on L-Systems”. In: *Plant and Soil* 332.1-2 (2010), pp. 177–192. DOI: 10.1007/s11104-010-0284-7.

[103] D. Leitner et al. “The algorithmic beauty of plant roots – an L-System model for dynamic root growth simulation”. In: *Mathematical and Computer Modelling of Dynamical Systems* 16.6 (Dec. 2010), pp. 575–587. DOI: 10.1080/13873954.2010.491360.

[104] R. Leuning. “A critical appraisal of a combined stomatal-photosynthesis model for C3 plants”. In: *Plant, Cell & Environment* 18.4 (1995), pp. 339–355. DOI: 10.1111/j.1365-3040.1995.tb00370.x.

[105] P. Levy. “Biomass expansion factors and root : shoot ratios for coniferous tree species in Great Britain”. In: *Forestry* 77.5 (May 2004), pp. 421–430. DOI: 10.1093/forestry/77.5.421.

[106] Y.-S. Lin, B. E. Medlyn, and D. S. Ellsworth. “Temperature responses of leaf net photosynthesis: the role of component processes”. In: *Tree physiology* 32.2 (2012), pp. 219–231. DOI: 10.1093/treephys/tpr141.

[107] S. P. Long. “Modification of the response of photosynthetic productivity to rising temperature by atmospheric CO2 concentrations: Has its importance been underestimated?” In: *Plant, Cell and Environment* 14.8 (Oct. 1991), pp. 729–739. DOI: 10.1111/j.1365-3040.1991.tb01439.x.

[108] J. López-Bucio, A. Cruz-Ramırez, and L. Herrera-Estrella. “The role of nutrient availability in regulating root architecture”. In: *Current opinion in plant biology* 6.3 (2003), pp. 280–287. DOI: 10.1016/S1369-5266(03)00035-9.

[109] B. Loveys. “The intracellular location of abscisic acid in stressed and non-stressed leaf tissue”. In: *Physiologia plantarum* 40.1 (1977), pp. 6–10. DOI: 10.1111/j.1399-3054.1977.tb01483.x.

[110] D. R. Lungley. “The growth of root systems - A numerical computer simulation model”. In: *Plant and Soil* 38.1 (Feb. 1973), pp. 145–159. DOI: 10.1007/bf00011223.

[111] J. Lynch. “Root Architecture and Plant Productivity”. In: *Plant Physiology* 109.1 (Sept. 1995), pp. 7–13. DOI: 10.1104/pp.109.1.7.

[112] J. P. Lynch. “Steep, cheap and deep: an ideotype to optimize water and N acquisition by maize root systems”. In: *Annals of Botany* 112.2 (Jan. 2013), pp. 347–357. DOI: 10.1093/aob/mcs293.

[113] J. Lynch. “Root Architecture and Nutrient Acquisition”. In: *Nutrient Acquisition by Plants: An Ecological Perspective*. Ed. by H. BassiriRad. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 147–183. ISBN: 978-3-540-27675-3. DOI: 10.1007/3-540-27675-0_7.

[114] J. P. Lynch. “Rightsizing Root Phenotypes for Drought resistance”. In: *Journal of experimental botany* (2018). DOI: 10.1093/jxb/ery048.

[115] J. P. Lynch. "Root phenotypes for improved nutrient capture: an underexploited opportunity for global agriculture". In: *New Phytologist* (2019). DOI: 10.1111/nph.15738.

[116] J. P. Lynch and K. M. Brown. "Topsoil foraging–an architectural adaptation of plants to low phosphorus availability". In: *Plant and Soil* 237.2 (2001), pp. 225–237. DOI: 10.1023/A:1013324727040.

[117] J. P. Lynch, M. D. Ho, et al. "Rhizoeconomics: carbon costs of phosphorus acquisition". In: *Plant and Soil* 269.1-2 (2005), pp. 45–56. DOI: 10.1007/s11104-004-1096-4.

[118] J. P. Lynch and J. Postma. "Invited Talk: Structural-Functional Model SimRoot and its Applications". In: *Plant Growth Modeling, Simulation, Visualization and Applications (PMA), 2009 Third International Symposium on*. IEEE. 2009, pp. 125–125. DOI: 10.1109/PMA.2009.73.

[119] J. P. Lynch et al. "SimRoot: Modelling and visualization of root systems". In: *Plant and Soil* 188.1 (1997), pp. 139–151. DOI: 10.1023/a:1004276724310.

[120] R. Madi et al. "Parametric soil water retention models: a critical evaluation of expressions for the full moisture range". In: *Hydrology and Earth System Sciences* 22.2 (Feb. 2018), pp. 1193–1219. DOI: 10.5194/hess-22-1193-2018.

[121] S. Mairhofer et al. "RooTrak: Automated Recovery of Three-Dimensional Plant Root Architecture in Soil from X-Ray Microcomputed Tomography Images Using Visual Tracking". In: *Plant Physiology* 158.2 (Dec. 2011), pp. 561–569. DOI: 10.1104/pp.111.186221.

[122] H. Majdi. "Root sampling methods - applications and limitations of the minirhizotron technique". In: *Plant and Soil* 185.2 (Sept. 1996), pp. 255–258. DOI: 10.1007/bf02257530.

[123] H. Majdi and P. Kangas. "Demography of fine roots in response to nutrient applications in a Norway spruce stand in southwestern Sweden". In: *Écoscience* 4.2 (Jan. 1997), pp. 199–205. DOI: 10.1080/11956860.1997.11682396.

[124] N. Malajczuk and K. Cromack. “Accumulation of Calcium Oxalate in the Mantle of Ectomycorrhizal Roots of Pinus Radiata and Eucalyptus Marginata”. In: *New Phytologist* 92.4 (Dec. 1982), pp. 527–531. DOI: 10.1111/j.1469-8137.1982.tb03411.x.

[125] M. Manzi et al. “Root ABA Accumulation in Long-Term Water-Stressed Plants is Sustained by Hormone Transport from Aerial Organs”. In: *Plant and Cell Physiology* 56.12 (Nov. 2015), pp. 2457–2466. DOI: 10.1093/pcp/pcv161.

[126] S. A. McAdam and T. J. Brodribb. “Linking Turgor with ABA Biosynthesis: Implications for Stomatal Responses to Vapor Pressure Deficit across Land Plants”. In: *Plant Physiology* 171.3 (May 2016), pp. 2008–2016. DOI: 10.1104/pp.16.00380.

[127] B. M. McKenzie et al. “Root-soil friction: quantification provides evidence for measurable benefits for manipulation of root-tip traits”. In: *Plant, Cell & Environment* 36.6 (Dec. 2012), pp. 1085–1092. DOI: 10.1111/pce.12037.

[128] R. E. McMurtrie and Y.-P. Wang. “Mathematical models of the photosynthetic response of tree stands to rising CO2 concentrations and temperatures”. In: *Plant, Cell and Environment* 16.1 (Jan. 1993), pp. 1–13. DOI: 10.1111/j.1365-3040.1993.tb00839.x.

[129] C. A. Micchelli, Y. Xu, and H. Zhang. “Universal kernels”. English. In: *Journal of Machine Learning Research (JMLR)* 7 (2006), pp. 2651–2667. ISSN: 1532-4435.

[130] J. J. Michalsky. “The astronomical almanac’s algorithm for approximate solar position (1950–2050)”. In: *Solar energy* 40.3 (1988), pp. 227–235. DOI: 10.1016/0038-092X(88)90045-X.

[131] G. S. L. Miner. “Measuring and modeling transpiration and photosynthesis in Zea mays and Helianthus annuus: leaf-level and sap flow studies”. In: *2000-2019-CSU Theses and Dissertations* (2016).

[132] G. N. Mitra. “Regulation of nutrient uptake by plants”. In: *New Delhi: Springer* 10 (2015), pp. 978–981.

[133] J. Monteith. "Evaporation and surface temperature". In: *Quarterly Journal of the Royal Meteorological Society* 107.451 (1981), pp. 1–27. DOI: 10.1002/qj.49710745102.

[134] J. L. Monteith. "Evaporation and environment". In: *Symp. Soc. Exp. Biol.* Vol. 19. 1965, p. 4.

[135] S. J. Mooney et al. "Developing X-ray Computed Tomography to non-invasively image 3-D root systems architecture in soil". In: *Plant and Soil* 352.1-2 (Nov. 2011), pp. 1–22. DOI: 10.1007/s11104-011-1039-9.

[136] A. Morales et al. "Dynamic modelling of limitations on improving leaf CO2 assimilation under fluctuating irradiance". In: *Plant, cell & environment* 41.3 (2018), pp. 589–604. DOI: 10.1111/pce.13119.

[137] K. K. Newsham, A. H. Fitter, and A. R. Watkinson. "Arbuscular Mycorrhiza Protect an Annual Grass from Root Pathogenic Fungi in the Field". In: *The Journal of Ecology* 83.6 (Dec. 1995), p. 991. DOI: 10.2307/2261180.

[138] K. L. Nielsen, A. Eshel, and J. P. Lynch. "The effect of phosphorus availability on the carbon economy of contrasting common bean (Phaseolus vulgaris L.) genotypes". In: *Journal of experimental botany* 52.355 (2001), pp. 329–339. DOI: 10.1093/jexbot/52.355.329.

[139] K. L. Nielsen et al. "Carbon cost of root systems: an architectural approach". In: *Plant and Soil* 165.1 (Mar. 1994), pp. 161–169. DOI: 10.1007/bf00009972.

[140] P. S. Nobel, D. M. Alm, and J. Cavelier. "Growth Respiration, Maintenance Respiration and Structural-Carbon Costs for Roots of Three Desert Succulents". In: *Functional Ecology* 6.1 (1992), p. 79. DOI: 10.2307/2389774.

[141] P. S. Nobel. *Physicochemical and Environmental Plant Physiology*. Academic Press, 2009.

[142] K. F. Nwanze. "Relationships between cassava root yields and crop infestations by the mealybug,Phenacoccus manihoti". In: *Tropical Pest Management* 28.1 (Mar. 1982), pp. 27–32. DOI: 10.1080/09670878209370669.

[143] Y. Ohwaki and H. Hirata. “Differences in carboxylic acid exudation among p-starved leguminous crops in relation to carboxylic acid contents in plant tissues and phospholipid level in roots”. In: *Soil Science and Plant Nutrition* 38.2 (June 1992), pp. 235–243. DOI: 10.1080/00380768.1992.10416486.

[144] M. Okamoto et al. “High Humidity Induces Abscisic Acid 8’-Hydroxylase in Stomata and Vasculature to Regulate Local and Systemic Abscisic Acid Responses in Arabidopsis”. In: *Plant Physiology* 149.2 (Nov. 2008), pp. 825–834. DOI: 10.1104/pp.108.130823.

[145] L. Pagès, M. O. Jordan, and D. Picard. “A simulation model of the three-dimensional architecture of the maize root system”. In: *Plant and Soil* 119.1 (Sept. 1989), pp. 147–154. DOI: 10.1007/bf02370279.

[146] L. Pagès et al. “Root Typ: a generic model to depict and analyse the root system architecture”. In: *Plant and soil* 258.1 (2004), pp. 103–119. DOI: 10.1023/B:PLSO.0000016540.47134.03.

[147] H. L. Penman. “Natural evaporation from open water, bare soil and grass”. In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*. Vol. 193. The Royal Society, 1948, pp. 120–145. DOI: 10.1098/rspa.1948.0037.

[148] R. van der Ploeg, W. Böhm, and M. Kirkham. “On the Origin of the Theory of Mineral Nutrition of Plants and the Law of the Minimum”. In: *Soil Science Society of America Journal* 63.5 (Sept. 1999), pp. 1055–1062. DOI: 10.2136/sssaj1999.6351055x.

[149] J. Polomski et al. “Root research methods”. In: *Plant Roots Hidden Half. 3rd ed. Marcel Dekker NY Pub* (2002), pp. 447–88.

[150] H. Poorter, C. Remkes, and H. Lambers. “Carbon and Nitrogen Economy of 24 Wild Species Differing in Relative Growth Rate”. In: *Plant Physiology* 94.2 (Oct. 1990), pp. 621–627. DOI: 10.1104/pp.94.2.621.

[151] A. de la Porte et al. “A Gaseous Milieu: Extending the Boundaries of the Rhizosphere”. In: *Trends in Microbiology* 28.7 (July 2020), pp. 536–542. DOI: 10.1016/j.tim.2020.02.016.

[152] J. R. Porter, B. Klepper, and R. K. Belford. "A model (WHTROOT) which synchronizes root growth and development with shoot development for winter wheat". In: *Plant and Soil* 92.1 (Feb. 1986), pp. 133–145. DOI: 10.1007/bf02372274.

[153] W. M. Post et al. "Soil carbon pools and world life zones". In: *Nature* 298.5870 (July 1982), pp. 156–159. DOI: 10.1038/298156a0.

[154] J. A. Postma, A. Dathe, and J. P. Lynch. "The Optimal Lateral Root Branching Density for Maize Depends on Nitrogen and Phosphorus Availability". In: *Plant Physiology* 166.2 (May 2014), pp. 590–602. DOI: 10.1104/pp.113.233916.

[155] J. A. Postma and J. P. Lynch. "Complementarity in root architecture for nutrient uptake in ancient maize/bean and maize/bean/squash polycultures". In: *Annals of Botany* 110.2 (Apr. 2012), pp. 521–534. DOI: 10.1093/aob/mcs082.

[156] J. A. Postma et al. "OpenSimRoot: widening the scope and application of root architectural models". In: *New Phytologist* (2017). DOI: 10.1111/nph.14641.

[157] J. A. Postma and C. K. Black. "Advances in root architectural modeling". In: *Understanding and improving crop root function*. Burleigh Dodds Science Publishing, Jan. 2021, pp. 3–32. DOI: 10.19103/as.2020.0075.02.

[158] J. A. Postma and J. P. Lynch. "Theoretical evidence for the functional benefit of root cortical aerenchyma in soils with low phosphorus availability". In: *Annals of Botany* 107.5 (Oct. 2010), pp. 829–841. ISSN: 0305-7364. DOI: 10.1093/aob/mcq199.

[159] J. A. Postma and J. P. Lynch. "Root cortical aerenchyma enhances the growth of maize on soils with suboptimal availability of nitrogen, phosphorus, and potassium". In: *Plant physiology* 156.3 (2011), pp. 1190–1201. DOI: 10.1104/pp.111.175489.

[160] K. S. Pregitzer, R. L. Hendrick, and R. Fogel. "The demography of fine roots in response to patches of water and nitrogen". In: *New Phytologist* 125.3 (Nov. 1993), pp. 575–580. DOI: 10.1111/j.1469-8137.1993.tb03905.x.

[161] K. S. Pregitzer et al. “Atmospheric CO2, soil nitrogen and turnover of fine roots”. In: *New Phytologist* 129.4 (Apr. 1995), pp. 579–585. DOI: 10.1111/j.1469-8137.1995.tb03025.x.

[162] K. S. Pregitzer et al. “Interactive Effects of Atmospheric CO2 and Soil-N Availability on Fine Roots of Populus tremuloides”. In: *Ecological Applications* 10.1 (Feb. 2000), pp. 18–33. DOI: 10.1890/1051-0761(2000)010[0018:ieoaca]2.0.co;2.

[163] F. Pukelsheim. “The Three Sigma Rule”. In: *The American Statistician* 48.2 (May 1994), p. 88. DOI: 10.2307/2684253.

[164] H. Rangarajan, J. A. Postma, and J. P. Lynch. “Co-optimization of axial root phenotypes for nitrogen and phosphorus acquisition in common bean”. In: *Annals of botany* 122.3 (2018), pp. 485–499. DOI: 10.1093/aob/mcy092.

[165] D. P. Rasse, C. Rumpel, and M.-F. Dignac. “Is soil carbon mostly root carbon? Mechanisms for a specific stabilisation”. In: *Plant and Soil* 269.1-2 (Feb. 2005), pp. 341–356. DOI: 10.1007/s11104-004-0907-y.

[166] P. Rekacewicz. *World Atlas of Desertification*. UNEP/GRID-Arendal. UNEP, International Soil Reference and Information Centre (ISRIC), 1997.

[167] H. Ren et al. “Dynamic analysis of ABA accumulation in relation to the rate of ABA catabolism in maize tissues under water deficit”. In: *Journal of Experimental Botany* 58.2 (Nov. 2006), pp. 211–219. DOI: 10.1093/jxb/erl117.

[168] C. J. Rhodes. “Soil erosion, climate change and global food security: challenges and strategies”. In: *Science progress* 97.2 (2014), pp. 97–153. DOI: 10.3184/003685014X13994567941465.

[169] J. H. Richards and M. M. Caldwell. “Hydraulic lift: substantial nocturnal water transport between soil layers by Artemisia tridentata roots”. In: *Oecologia* 73.4 (1987), pp. 486–489. DOI: 10.1007/BF00379405.

[170] D. Robinson et al. “Plant root proliferation in nitrogen-rich patches confers competitive advantage”. In: *Proceedings of the Royal Society B: Biological Sciences* 266.1418 (Mar. 1999), pp. 431–435. DOI: 10.1098/rspb.1999.0656.

[171] D. Robinson. "The responses of plants to non-uniform supplies of nutrients". In: *New Phytologist* 127.4 (Aug. 1994), pp. 635–674. DOI: 10.1111/j.1469-8137.1994.tb02969.x.

[172] G. Rubio et al. "Topsoil foraging and its role in plant competitiveness for phosphorus in common bean". In: *Crop Science* 43.2 (2003), pp. 598–607. DOI: 10.2135/cropsci2003.5980.

[173] L. Sack, G. P. John, and T. N. Buckley. "ABA Accumulation in Dehydrating Leaves Is Associated with Decline in Cell Volume, Not Turgor Pressure". In: *Plant Physiology* 176.1 (Oct. 2017), pp. 489–495. DOI: 10.1104/pp.17.01097.

[174] P. Saengwilai et al. "Root cortical aerenchyma enhances nitrogen acquisition from low-nitrogen soils in maize". In: *Plant physiology* 166.2 (2014), pp. 726–735. DOI: 10.1104/pp.114.241711.

[175] H. Schneider et al. "Root Cortical Senescence Improves Growth under Suboptimal Availability of N, P, and K". In: *Plant physiology* (2017), pp–00648. DOI: 10.1104/pp.17.00648.

[176] A. Schnepf, D. Leitner, and S. Klepsch. "Modeling Phosphorus Uptake by a Growing and Exuding Root System". In: *Vadose Zone Journal* 11.3 (Aug. 2012), vzj2012.0001. DOI: 10.2136/vzj2012.0001.

[177] A. Schnepf et al. "Modelling Phosphorus Dynamics in the Soil–Plant System". In: *Soil Biology*. Springer Berlin Heidelberg, Oct. 2010, pp. 113–133. DOI: 10.1007/978-3-642-15271-9_5.

[178] G. Schwarz et al. "Estimating the dimension of a model". In: *The annals of statistics* 6.2 (1978), pp. 461–464. URL: https://www.jstor.org/stable/2958889.

[179] J. Simunek, K. Huang, and M. T. Van Genuchten. "The SWMS_3D code for simulating water flow and solute transport in three-dimensional variably-saturated media". In: *US Salinity Laboratory Research Report* 139 (1995).

[180] S. S. SNAPP and C. Shennan. "Effects of salinity on root growth and death dynamics of tomato, Lycopersicon esculentum Mill." In: *New Phytologist* 121.1 (May 1992), pp. 71–79. DOI: 10.1111/j.1469-8137.1992.tb01094.x.

[181] L. Y. Spek. In: *Plant and Soil* 197.1 (1997), pp. 9–18. DOI: 10.1023/a:1004236626479.

[182] C. J. T. Spitters and A. H. C. M. Schapendonk. "Evaluation of breeding strategies for drought tolerance in potato by means of crop growth simulation". In: *Genetic Aspects of Plant Mineral Nutrition*. Springer Netherlands, 1990, pp. 151–161. DOI: 10.1007/978-94-009-2053-8_24.

[183] S. J. Steele et al. "Root mass, net primary production and turnover in aspen, jack pine and black spruce forests in Saskatchewan and Manitoba, Canada". In: *Tree physiology* 17.8-9 (1997), pp. 577–587. DOI: 10.1093/treephys/17.8-9.577.

[184] B. Steingrobe, H. Schmid, and N. Claassen. "Root production and root mortality of winter barley and its implication with regard to phosphate acquisition". In: *Plant and Soil* 237.2 (2001), pp. 239–248. DOI: 10.1023/A:1013345718414.

[185] B. Steingrobe. "Root renewal of sugar beet as a mechanism of P uptake efficiency". In: *Journal of Plant Nutrition and Soil Science* 164.5 (2001), pp. 533–539. DOI: 10.1002/1522-2624(200110)164:5<533::AID-JPLN533>3.0.CO;2-D.

[186] B. Steingrobe. "A sensitivity analysis for assessing the relevance of fine-root turnover for P and K uptake". In: *Journal of Plant Nutrition and Soil Science* 168.4 (2005), pp. 496–502. DOI: 10.1002/jpln.200424107.

[187] B. Steingrobe. "Root turnover of faba beans (Vicia faba L.) and its interaction with P and K supply". In: *Journal of Plant Nutrition and Soil Science* 168.3 (2005), pp. 364–371. DOI: 10.1002/jpln.200420499.

[188] B. Steingrobe et al. "Root production and root mortality of winter wheat grown on sandy and loamy soils in different farming systems". In: *Biology and fertility of soils* 33.4 (2001), pp. 331–339. DOI: 10.1007/s003740000334.

[189] C. O. Stöckle, M. Donatelli, and R. Nelson. "CropSyst, a cropping systems simulation model". In: *European Journal of Agronomy* 18.3-4 (Jan. 2003), pp. 289–307. DOI: 10.1016/s1161-0301(02)00109-0.

[190] C. F. Strock, L. M. De La Riva, and J. P. Lynch. "Reduction in root secondary growth as a strategy for phosphorus acquisition". In: *Plant physiology* 176.1 (2018), pp. 691–703. DOI: 10.1104/pp.17.01583.

[191] F. C. Sussmilch, T. J. Brodribb, and S. A. M. McAdam. "Up-regulation of NCED3 and ABA biosynthesis occur within minutes of a decrease in leaf turgor but AHK1 is not required". In: *Journal of Experimental Botany* 68.11 (Apr. 2017), pp. 2913–2918. DOI: 10.1093/jxb/erx124.

[192] S. Trachsel et al. "Maize root growth angles become steeper under low N conditions". In: *Field Crops Research* 140 (Jan. 2013), pp. 18–31. DOI: 10.1016/j.fcr.2012.09.010.

[193] S. Trachsel et al. "Shovelomics: high throughput phenotyping of maize (Zea mays L.) root architecture in the field". In: *Plant and Soil* 341.1-2 (Nov. 2010), pp. 75–87. DOI: 10.1007/s11104-010-0623-8.

[194] M. Unkovich, J. Baldock, and M. Forbes. "Variability in Harvest Index of Grain Crops and Potential Significance for Carbon Accounting". In: *Advances in Agronomy*. Elsevier, 2010, pp. 173–219. DOI: 10.1016/s0065-2113(10)05005-4.

[195] C. Varlet-Grancher et al. "Efficience de la conversion de l'énergie solaire par un couvert végétal." In: *Acta Oecologica Oecologia Plantarum* 3.1 (1982), pp. 3–26.

[196] I. Vernon, M. Goldstein, and R. Bower. "Galaxy formation: Bayesian history matching for the observable universe". In: *Statistical science* (2014), pp. 81–90. DOI: 10.1214/12-STS412.

[197] I. Vernon et al. "Bayesian uncertainty analysis for complex systems biology models: emulation, global parameter searches and evaluation of gene functions". In: *BMC systems biology* 12.1 (2018), p. 1. DOI: 10.1186/s12918-017-0484-3.

[198] V. Vives-Peris et al. "Root exudates: from plant to rhizosphere and beyond". In: *Plant Cell Reports* 39.1 (July 2019), pp. 3–17. DOI: 10.1007/s00299-019-02447-5.

[199] K. A. Vogt and H. Persson. "Measuring growth and development of roots". In: *Techniques and approaches in forest tree ecophysiology* (1991), pp. 477–501.

[200] S. Von Caemmerer and G. D. Farquhar. "Some relationships between the biochemistry of photosynthesis and the gas exchange of leaves". In: *Planta* 153.4 (1981), pp. 376–387. DOI: 10.1007/BF00384257.

[201] S. Von Caemmerer. *Biochemical models of leaf photosynthesis*. Csiro publishing, 2000.

[202] S. Von Caemmerer. "Steady-state models of photosynthesis". In: *Plant, Cell & Environment* 36.9 (2013), pp. 1617–1630. DOI: 10.1111/pce.12098.

[203] J. Vos et al. "Functional–structural plant modelling: a new versatile tool in crop science". In: *Journal of Experimental Botany* 61.8 (Dec. 2009), pp. 2101–2115. DOI: 10.1093/jxb/erp345.

[204] A. S. Walcroft et al. "The response of photosynthetic model parameters to temperature and nitrogen concentration in Pinus radiata D. Don". In: *Plant, Cell and Environment* 20.11 (Nov. 1997), pp. 1338–1348. DOI: 10.1046/j.1365-3040.1997.d01-31.x.

[205] T. C. Walk, R. Jaramillo, and J. P. Lynch. "Architectural tradeoffs between adventitious and basal roots for phosphorus acquisition". In: *Plant and Soil* 279.1-2 (2006), pp. 347–366. DOI: 10.1007/s11104-005-0389-6.

[206] T. S. Walker. "Root Exudation and Rhizosphere Biology". In: *Plant Physiology* 132.1 (May 2003), pp. 44–51. DOI: 10.1104/pp.102.019661.

[207] C. Watson. "Environment-induced Modifications to Root Longevity in Lolium perenne and Trifolium repens". In: *Annals of Botany* 85.3 (Mar. 2000), pp. 397–401. DOI: 10.1006/anbo.1999.1048.

[208] C. E. Wells. "Advances in the fine root demography of woody species". PhD thesis. Pennsylvania State University, 1999.

[209] G. Weste. "Vegetation Changes Associated With Invasion by Phytophthora cinnamomi of Defined Plots in the Brisbane Ranges, Victoria, 1975-1985". In: *Australian Journal of Botany* 34.6 (1986), p. 633. DOI: 10.1071/bt9860633.

[210] D. Williamson et al. "History matching for exploring and reducing climate model parameter space using observations and a large perturbed physics ensemble". In: *Climate Dynamics* 41.7-8 (Aug. 2013), pp. 1703–1729. DOI: 10.1007/s00382-013-1896-4.

[211] L. Wu et al. “SPACSYS: integration of a 3D root architecture component to carbon, nitrogen and water cycling—model description”. In: *Ecological Modelling* 200.3-4 (2007), pp. 343–359. DOI: 10.1016/j.ecolmodel.2006.08.010.

[212] W. Xu et al. “Drought stress condition increases root to shoot ratio via alteration of carbohydrate partitioning and enzymatic activity in rice seedlings”. In: *Acta physiologiae plantarum* 37.2 (2015), pp. 1–11. DOI: 10.1007/s11738-014-1760-0.

[213] T. Yamauchi et al. “Aerenchyma formation in crop species: A review”. In: *Field Crops Research* 152 (Oct. 2013), pp. 8–16. DOI: 10.1016/j.fcr.2012.12.008.

[214] R. Yanai and D. Eissenstat. “Root Life Span, Efficiency, and Turnover”. In: *Plant Roots*. Informa UK Limited, Mar. 2002, pp. 221–238. DOI: 10.1201/9780203909423.ch13.

[215] H. S. Yang and B. H. Janssen. “A mono-component model of carbon mineralization with a dynamic rate constant”. In: *European Journal of Soil Science* 51.3 (Sept. 2000), pp. 517–529. DOI: 10.1046/j.1365-2389.2000.00319.x.

[216] X. Yin et al. “Temperature response of bundle-sheath conductance in maize leaves”. In: *Journal of experimental botany* 67.9 (2016), pp. 2699–2714. DOI: 10.1093/jxb/erw104.

[217] L. M. York. “Functional phenomics: an emerging field integrating high-throughput phenotyping, physiology, and bioinformatics”. In: *Journal of Experimental Botany* 70.2 (Oct. 2018), pp. 379–386. DOI: 10.1093/jxb/ery379.

[218] L. M. York, E. A. Nord, and J. P. Lynch. “Integration of root phenes for soil resource acquisition”. In: *Frontiers in Plant Science* 4 (2013). DOI: 10.3389/fpls.2013.00355.

[219] A. Zhan and J. P. Lynch. “Reduced frequency of lateral root branching improves N capture from low-N soils in maize”. In: *Journal of Experimental Botany* 66.7 (2015), pp. 2055–2065. DOI: 10.1093/jxb/erv007.

[220] A. Zhan, H. Schneider, and J. P. Lynch. “Reduced lateral root branching density improves drought tolerance in maize”. In: *Plant physiology* 168.4 (2015), pp. 1603–1615. DOI: 10.1104/pp.15.00187.

[221] J. Zhu and J. P. Lynch. “The contribution of lateral rooting to phosphorus acquisition efficiency in maize (Zea mays) seedlings”. In: *Functional Plant Biology* 31.10 (2004), p. 949. DOI: 10.1071/fp04046.

[222] E. L. Zvereva and M. V. Kozlov. “Sources of variation in plant responses to belowground insect herbivory: a meta-analysis”. In: *Oecologia* 169.2 (Dec. 2011), pp. 441–452. DOI: 10.1007/s00442-011-2210-y.

Appendix A

Root Loss Supplementary Material

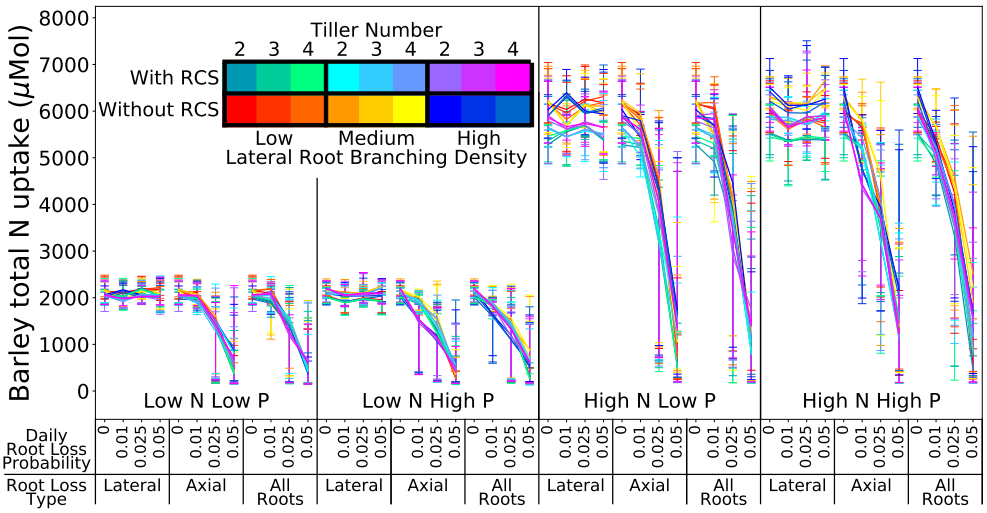


Figure A.1: Mean plant nitrogen uptake at 80 days of the 18 barley phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

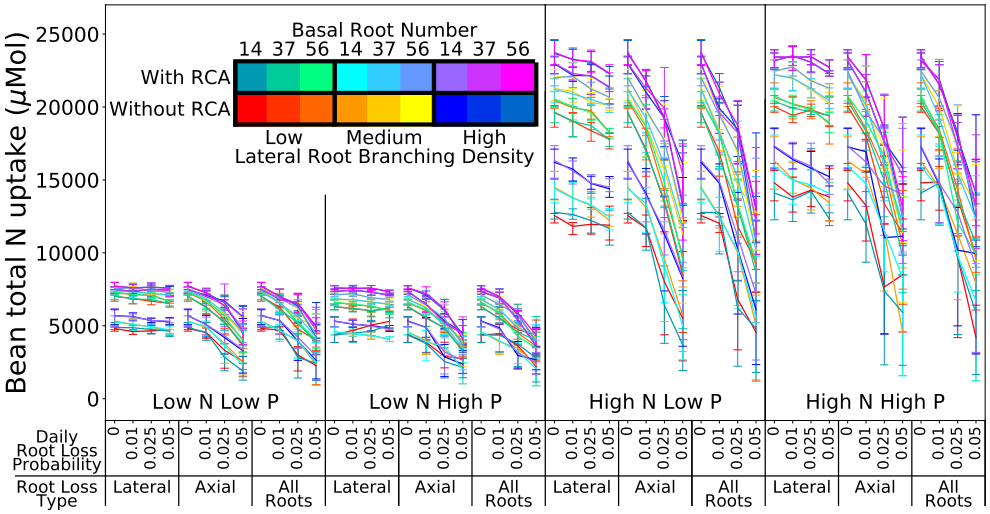


Figure A.2: Mean plant nitrogen uptake at 40 days of the 18 bean phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

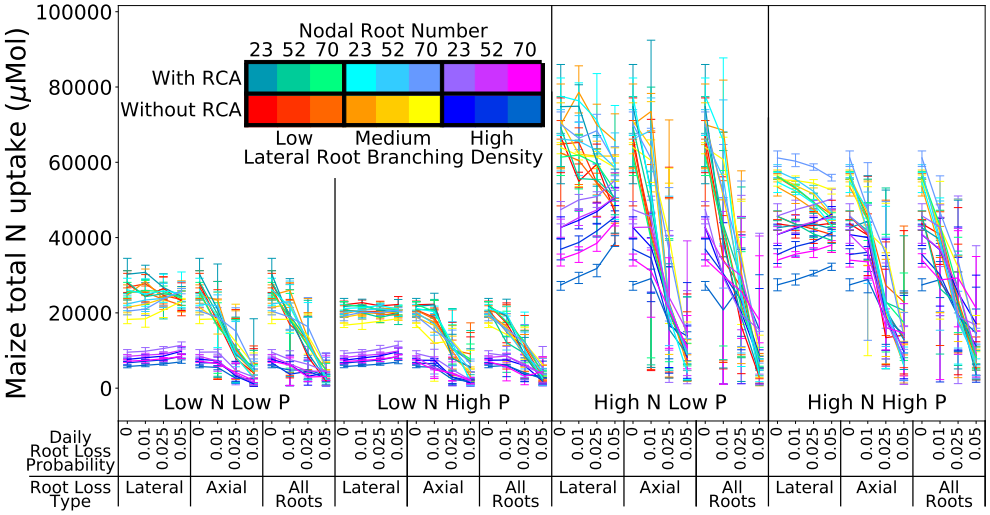


Figure A.3: Mean plant nitrogen uptake at 40 days of the 18 maize phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

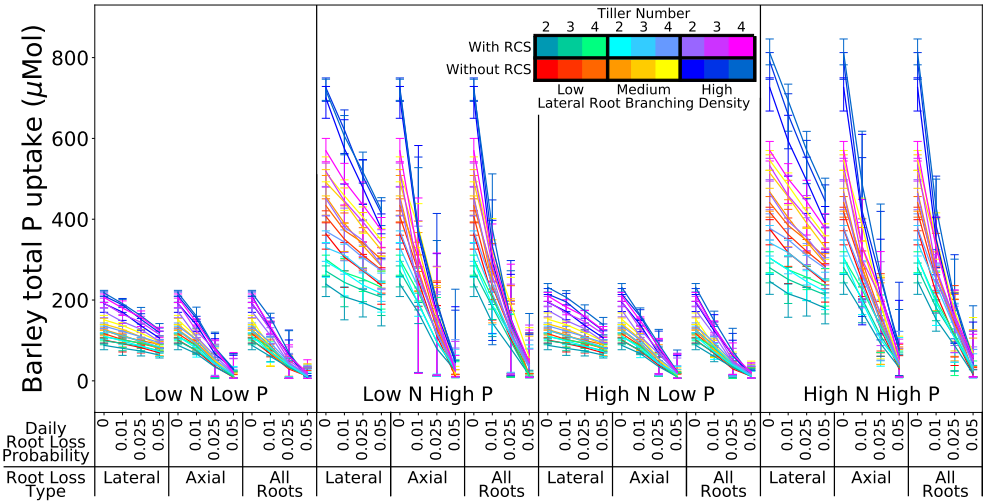


Figure A.4: Mean plant phosphorus uptake at 80 days of the 18 barley phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend at the top shows which colour corresponds to which phenotype.

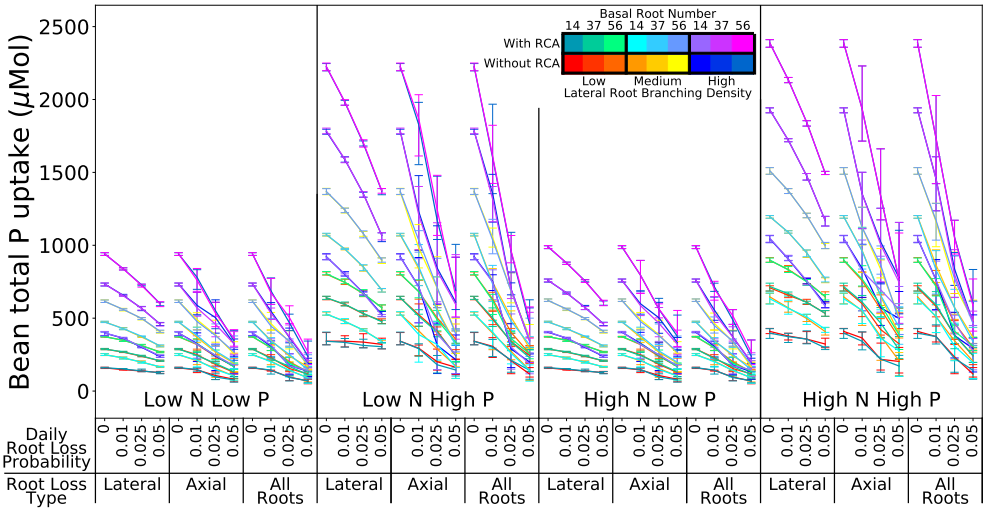


Figure A.5: Mean plant phosphorus uptake at 40 days of the 18 bean phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend at the top shows which colour corresponds to which phenotype.

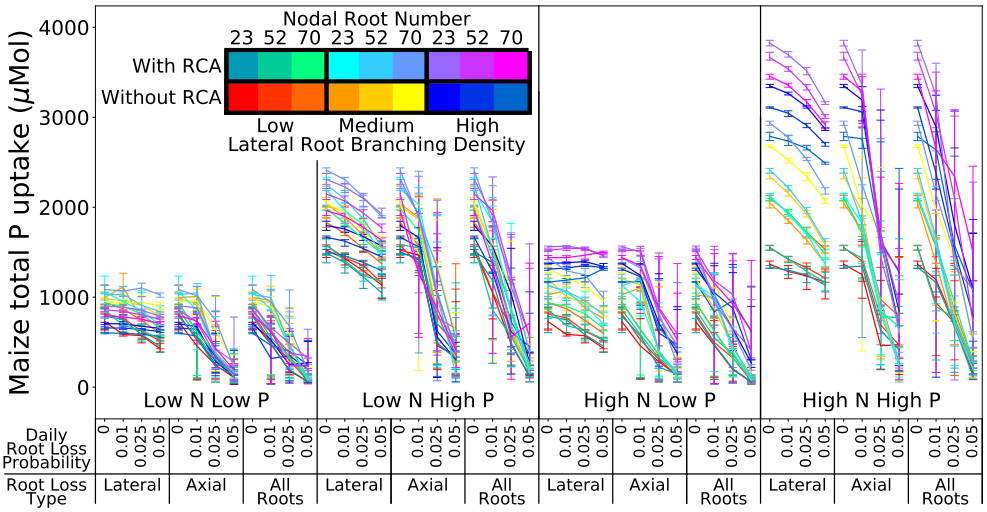


Figure A.6: Mean plant phosphorus uptake at 40 days of the 18 maize phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top shows which colour corresponds to which phenotype.

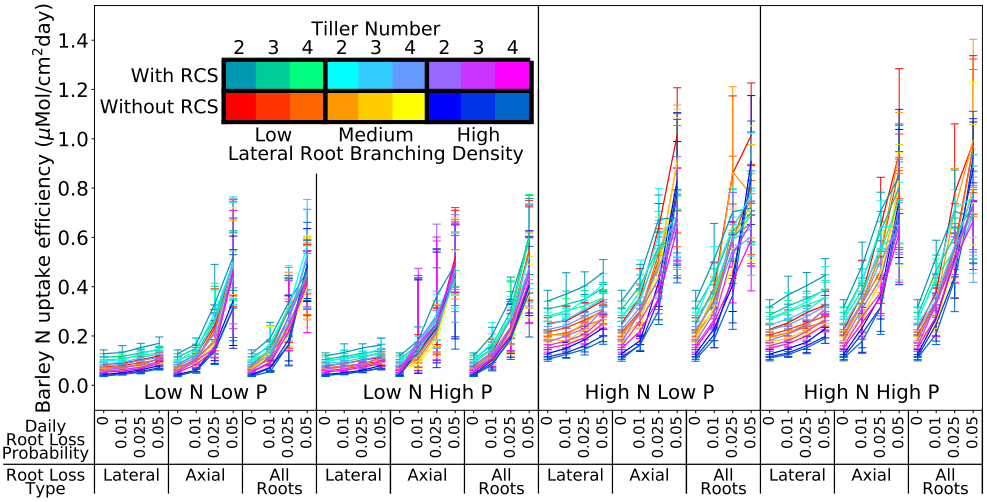


Figure A.7: Mean plant nitrogen uptake efficiency at 80 days of the 18 barley phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

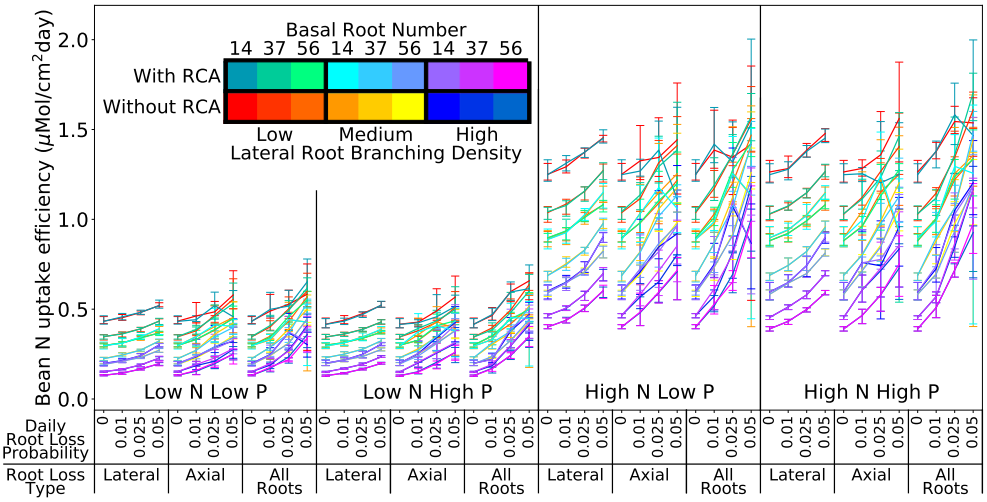


Figure A.8: Mean plant nitrogen uptake efficiency at 40 days of the 18 bean phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

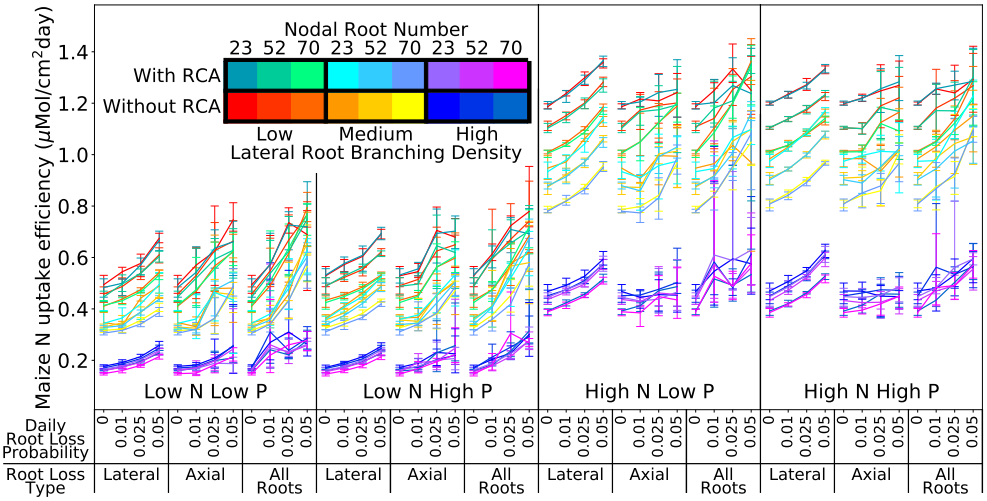


Figure A.9: Mean plant nitrogen uptake efficiency at 40 days of the 18 maize phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

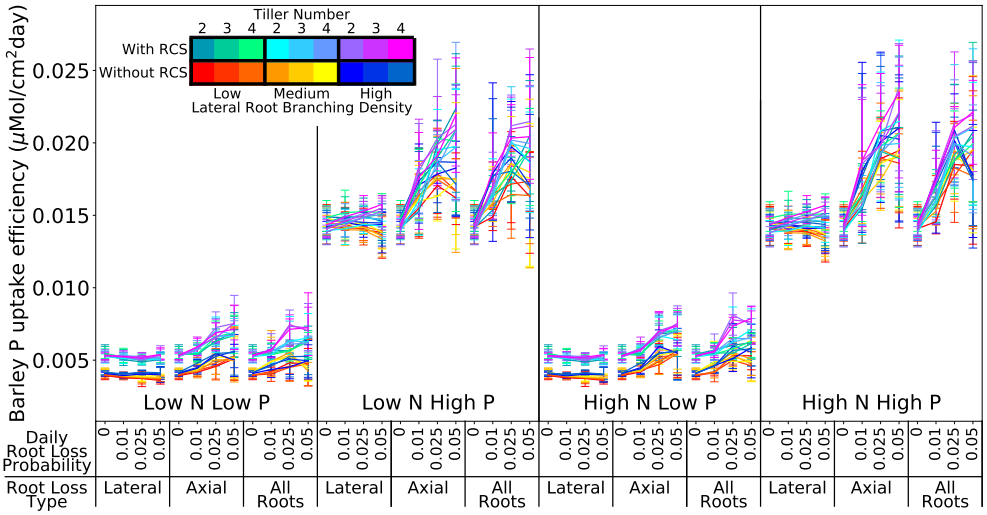


Figure A.10: Mean plant phosphorus uptake efficiency at 80 days of the 18 barley phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

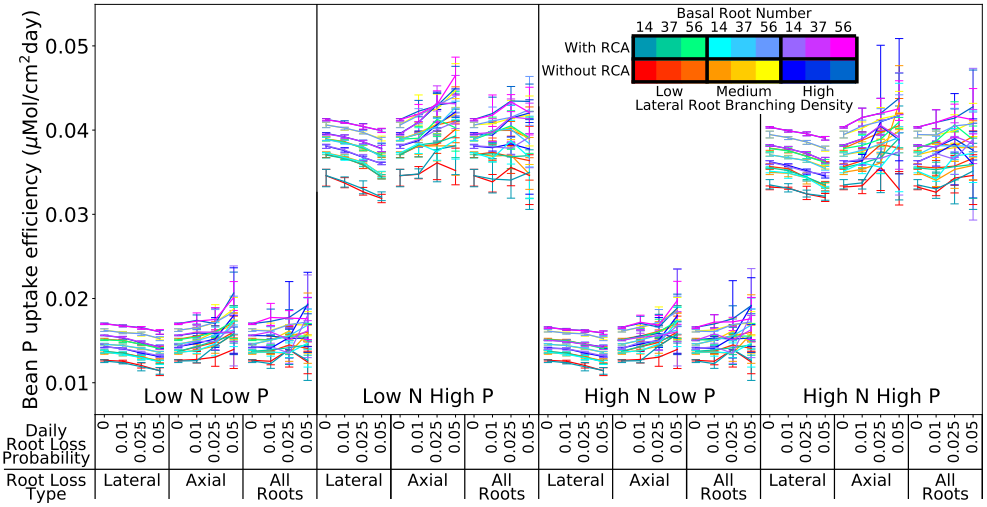


Figure A.11: Mean plant phosphorus uptake efficiency at 40 days of the 18 bean phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top right shows which colour corresponds to which phenotype.

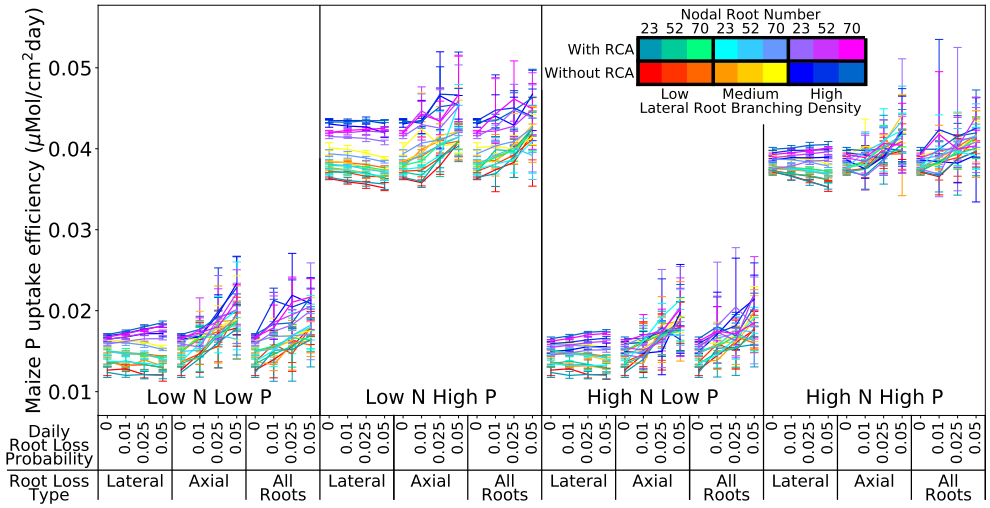


Figure A.12: Mean plant phosphorus uptake efficiency at 40 days of the 18 maize phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top right shows which colour corresponds to which phenotype.

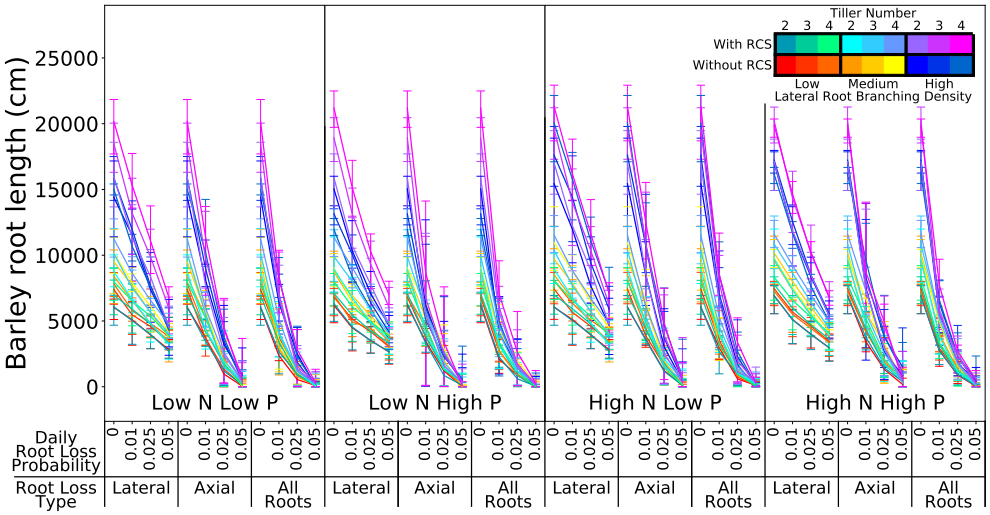


Figure A.13: Mean root length at 80 days of the 18 barley phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top right shows which colour corresponds to which phenotype.

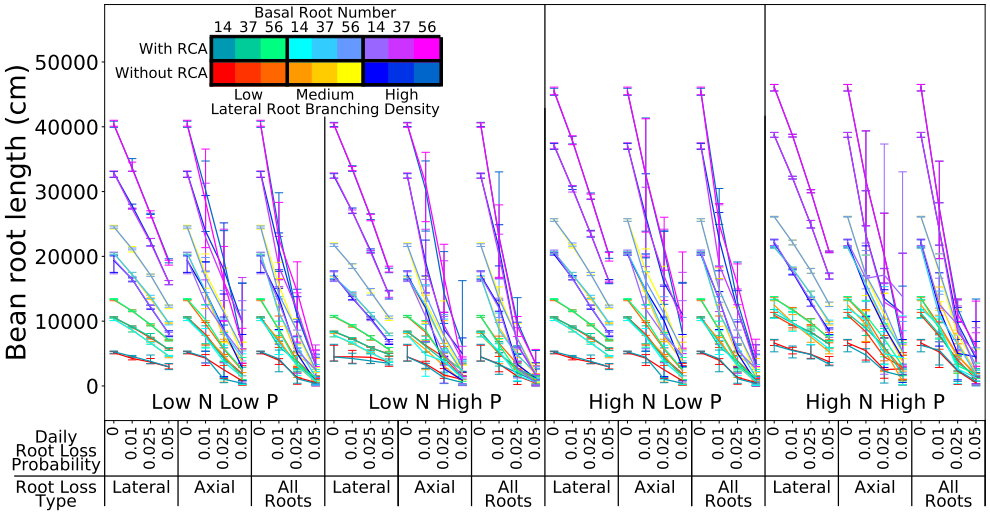


Figure A.14: Mean root length at 40 days of the 18 bean phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

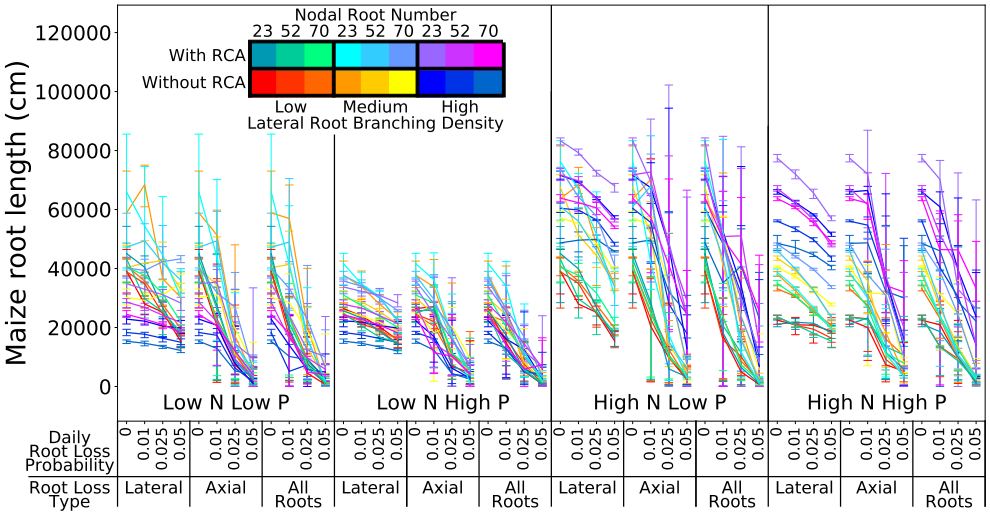


Figure A.15: Mean root length at 40 days of the 18 maize phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

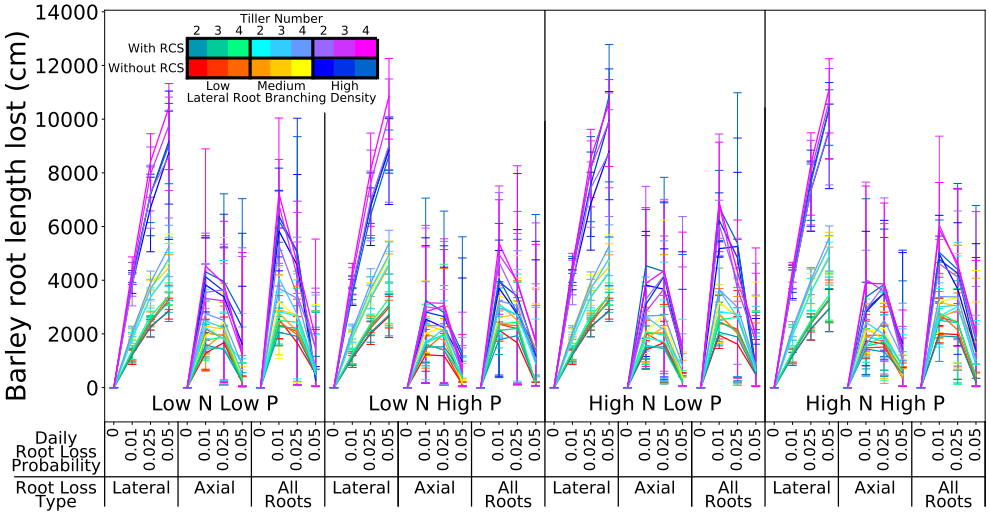


Figure A.16: Mean root length lost at 80 days of the 18 barley phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

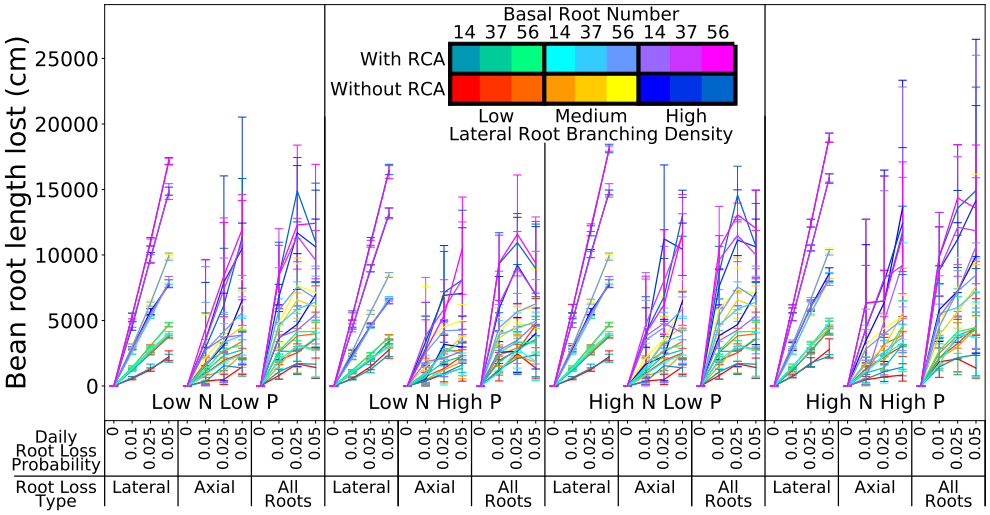


Figure A.17: Mean root length lost at 40 days of the 18 bean phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend at the top shows which colour corresponds to which phenotype.

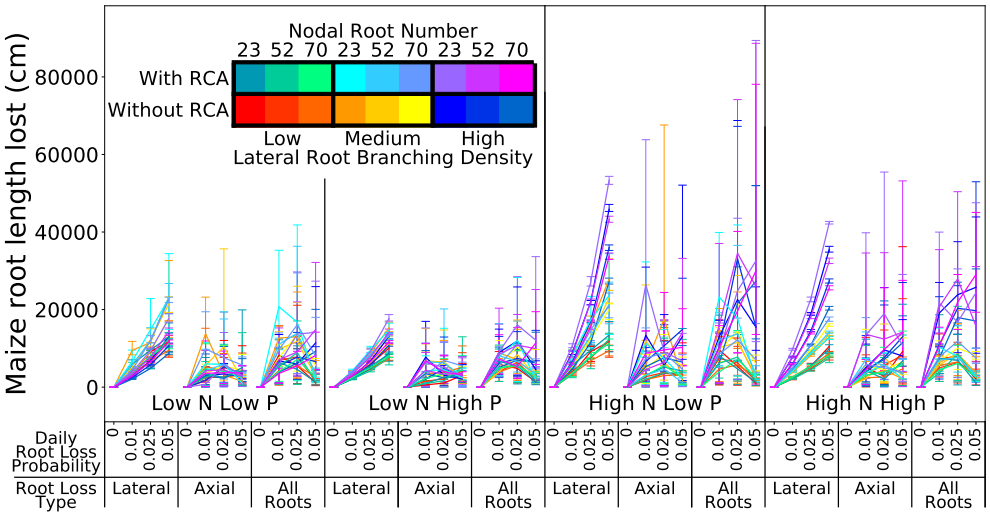


Figure A.18: Mean root length lost at 40 days of the 18 maize phenotypes under consideration, each in a different colour. The bars indicate minimum and maximum values. The figure is divided in 4 sections, each of which shows the results for one of the 4 different soils we considered. For every soil, we show the results for the three different types of root loss at 4 different intensities (which includes the case without any root loss). The legend in the top left shows which colour corresponds to which phenotype.

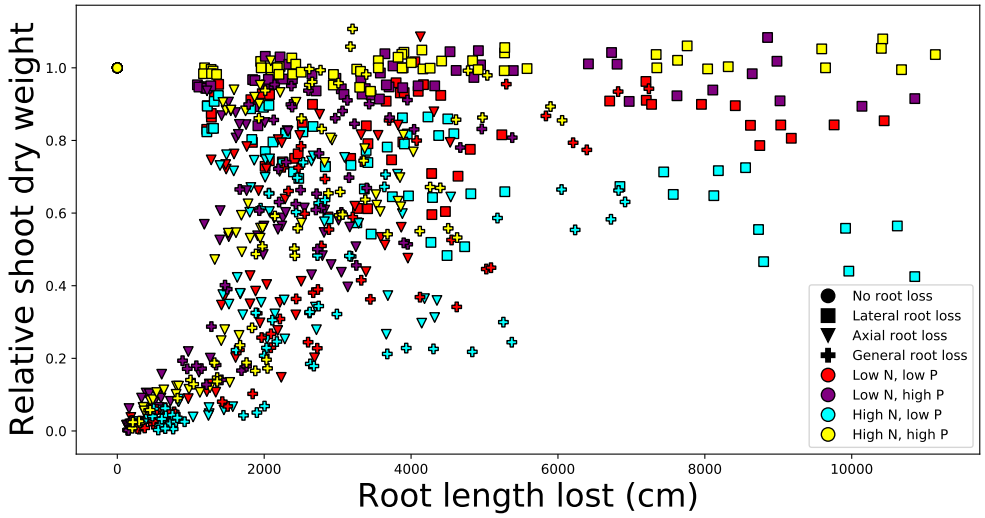


Figure A.19: Barley shoot dry weight relative to the reference case without root loss versus total root length lost after 80 days. Each point represents the mean of 5 repetitions, the shape of the point indicates the type of root loss and the colour of the point indicates the soil nutrient availability.

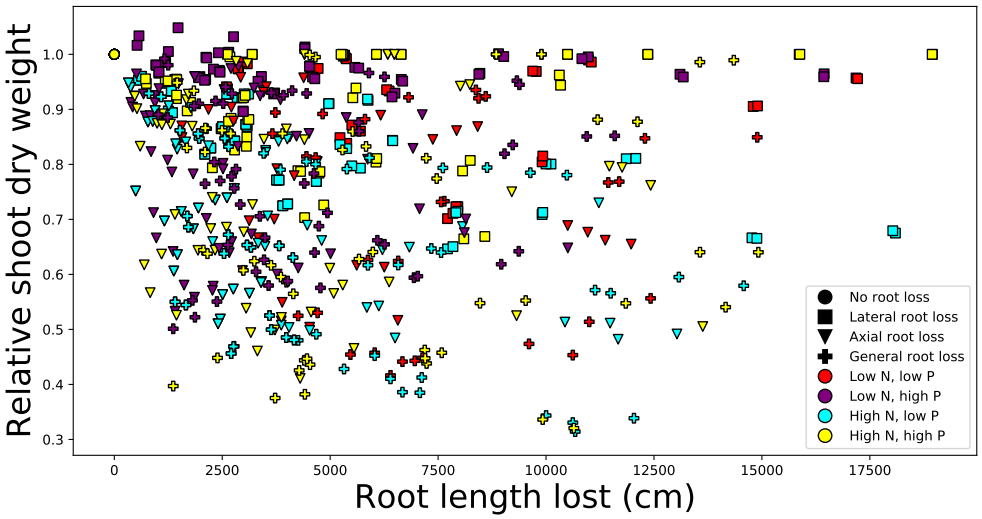


Figure A.20: Bean shoot dry weight relative to the reference case without root loss versus total root length lost after 40 days. Each point represents the mean of 5 repetitions, the shape of the point indicates the type of root loss and the colour of the point indicates the soil nutrient availability.

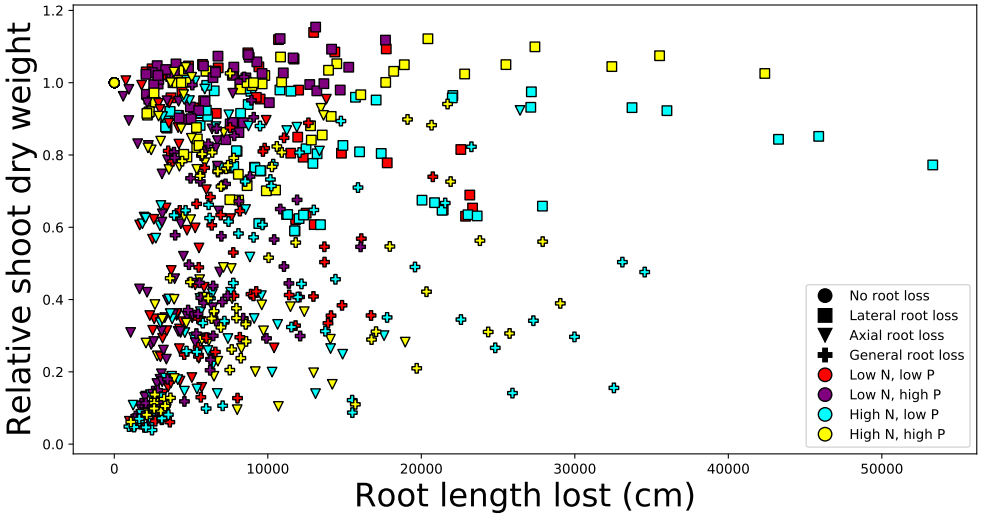


Figure A.21: Maize shoot dry weight relative to the reference case without root loss versus total root length lost after 40 days. Each point represents the mean of 5 repetitions, the shape of the point indicates the type of root loss and the colour of the point indicates the soil nutrient availability.

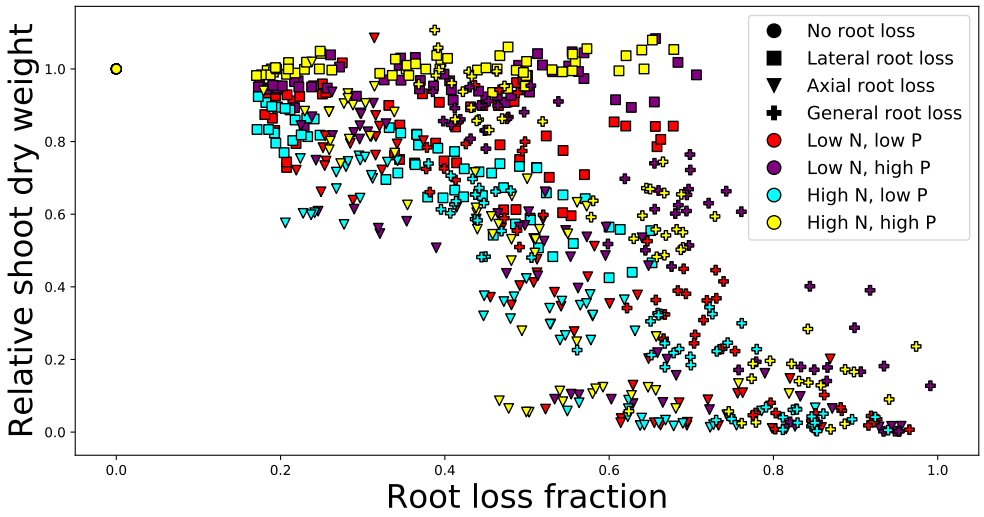


Figure A.22: Barley shoot dry weight relative to the reference case without root loss versus the root length lost as fraction of total root length produced after 80 days. Each point represents the mean of 5 repetitions, the shape of the point indicates the type of root loss and the colour of the point indicates the soil nutrient availability.

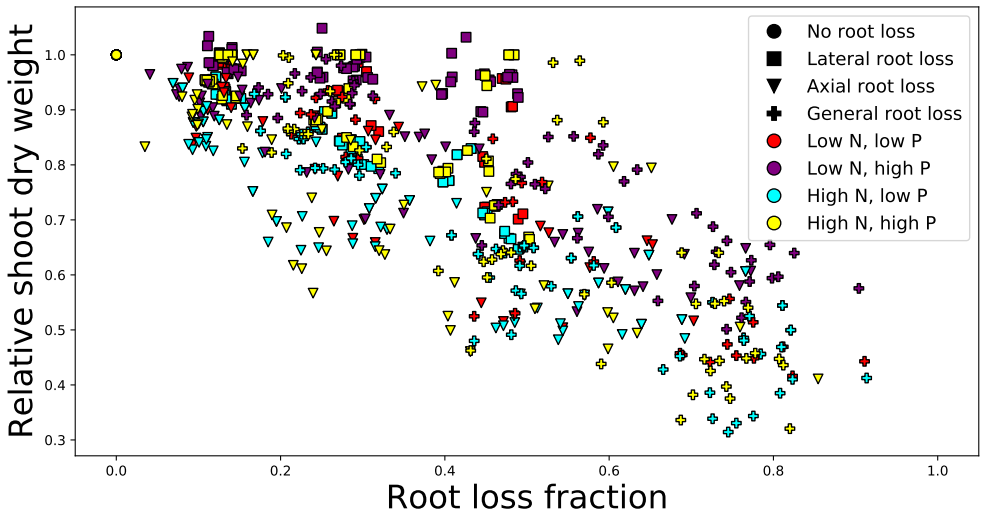


Figure A.23: Bean shoot dry weight relative to the reference case without root loss versus the root length lost as fraction of total root length produced after 40 days. Each point represents the mean of 5 repetitions, the shape of the point indicates the type of root loss and the colour of the point indicates the soil nutrient availability.

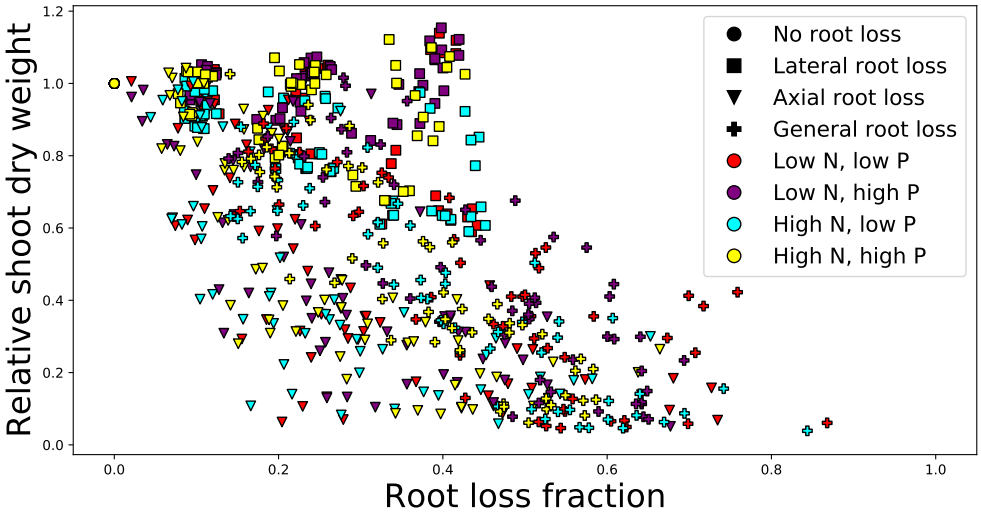


Figure A.24: Maize shoot dry weight relative to the reference case without root loss versus the root length lost as fraction of total root length produced after 40 days. Each point represents the mean of 5 repetitions, the shape of the point indicates the type of root loss and the colour of the point indicates the soil nutrient availability.

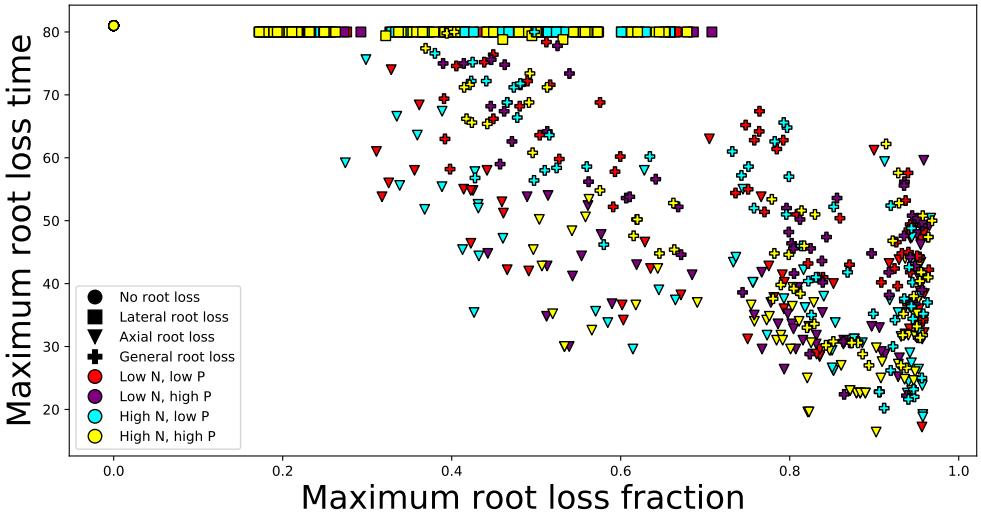


Figure A.25: The maximum root length lost, as fraction of total root length lost produced at that time, that was observed during the 80 days of simulation for barley, versus the time at which this maximum root loss fraction was observed. Each point represents the mean of 5 repetitions, the shape of the point indicates the type of root loss and the colour of the point indicates the soil nutrient availability.

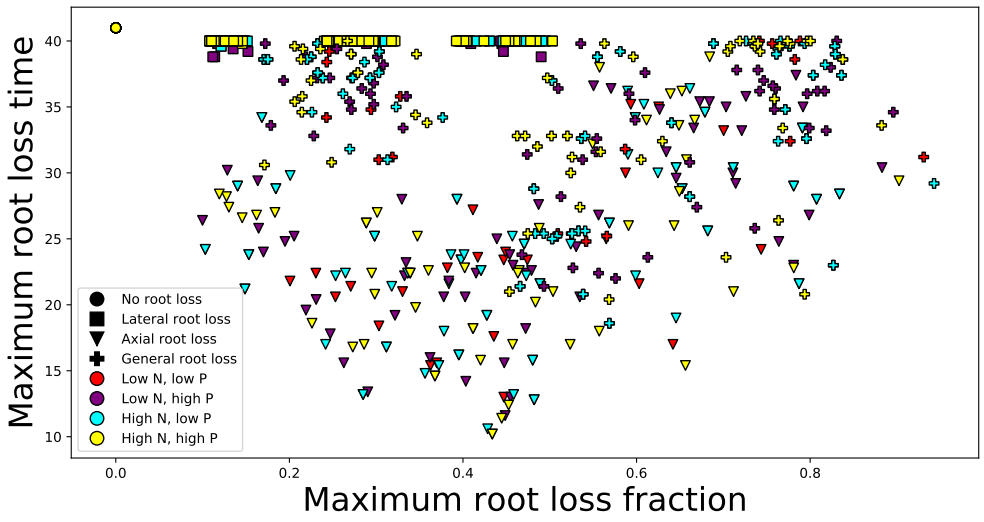


Figure A.26: The maximum root length lost, as fraction of total root length lost produced at that time, that was observed during the 40 days of simulation for bean, versus the time at which this maximum root loss fraction was observed. Each point represents the mean of 5 repetitions, the shape of the point indicates the type of root loss and the colour of the point indicates the soil nutrient availability.

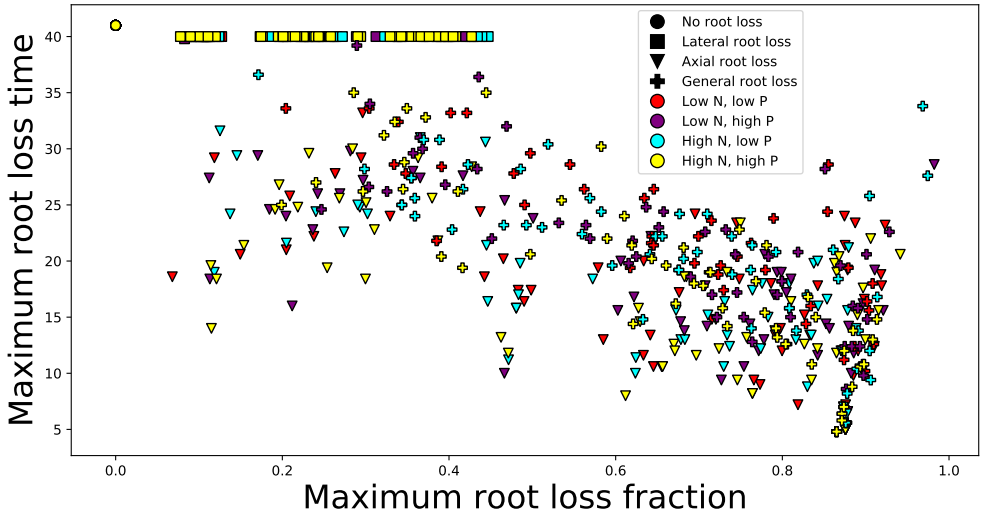


Figure A.27: The maximum root length lost, as fraction of total root length lost produced at that time, that was observed during the 40 days of simulation for maize, versus the time at which this maximum root loss fraction was observed. Each point represents the mean of 5 repetitions, the shape of the point indicates the type of root loss and the colour of the point indicates the soil nutrient availability.

Appendix B

Tables

B.1 Drought

Symbol	Variable	Equation	Unit
A	Photosynthesis rate	4.1.4, 4.5.3	$\frac{\mu mol}{m^2\,s}$
A_c	Carbon limited assimilation rate	4.1.5, 4.5.1	$\frac{\mu mol}{m^2\,s}$
A_j	Light limited assimilation rate	4.1.6, 4.5.2	$\frac{\mu mol}{m^2\,s}$
A_{IR}	Infrared irradiation leaf absorption	4.6.15	$\frac{W}{m^2\,s}$
A_S	Solar irradiation leaf absorption	4.6.14	$\frac{W}{m^2\,s}$
C_m	Mesophyll CO_2 concentration	4.1.11, 4.5.11	$\frac{\mu mol}{mol}$
C_s	Bundle sheath CO_2 concentration	4.5.12	$\frac{\mu mol}{mol}$
E	Leaf energy balance	4.6.25	$\frac{W}{m^2}$
e_{IR}	Leaf infrared radiation emission	4.6.18	$\frac{W}{m^2}$
g_w	Stomatal conductance	4.1.13	$\frac{mol}{m^2\,s}$
g_s	Bundle sheath conductance	4.6.12	$\frac{mol}{m^2\,s}$
H_c	Conduction and convection heat loss	4.6.19	$\frac{W}{m^2}$
H_t	Transpiration heat loss	4.6.23	$\frac{W}{m^2}$
H_{vap}	Latent heat of vaporisation	4.6.24	$\frac{J}{mol}$
I_2	Useful light absorbed by PS II	4.1.17	$\frac{\mu mol}{m^2\,s}$
I_s	Solar irradiation	4.7.1	$\frac{W}{m^2}$
J	Potential electron transport rate	4.1.16	$\frac{\mu mol}{m^2\,s}$
J_{max}	Maximum potential electron transport rate	4.6.10	$\frac{\mu mol}{m^2\,s}$

Table B.1: Derived quantities

Symbol	Variable	Equation	Unit
K_{air}	Air thermal conductivity	4.6.21	$\frac{\mathrm{W}}{\mathrm{m\,K}}$
K_C	CO_2 Michaelis constant	4.6.3	$\frac{\mu\mathrm{mol}}{\mathrm{mol}}$
K_O	O_2 Michaelis constant	4.6.4	$\frac{\mu\mathrm{mol}}{\mathrm{mol}}$
K_P	PEP Michaelis constant	4.6.7	$\frac{\mu\mathrm{mol}}{\mathrm{mol}}$
O_m	Mesophyll O_2 concentration	4.1.12, 4.5.13	$\frac{\mu\mathrm{mol}}{\mathrm{mol}}$
O_s	Bundle sheath O_2 concentration	4.5.14	$\frac{\mu\mathrm{mol}}{\mathrm{mol}}$
R_d	Dark respiration	4.6.5	$\frac{\mu\mathrm{mol}}{\mathrm{m^2\,s}}$
R_m	Mesophyll dark respiration	4.5.8	$\frac{\mu\mathrm{mol}}{\mathrm{m^2\,s}}$
R_s	Bundle sheath dark respiration	4.5.9	$\frac{\mu\mathrm{mol}}{\mathrm{m^2\,s}}$
$S_{c/o}$	Rubisco specificity	4.6.8	-
S_w	Water stress factor	4.1.1	-
V_{cmax}	Maximum rubisco carboxylation	4.6.6	$\frac{\mu\mathrm{mol}}{\mathrm{m^2\,s}}$
V_{pmax}	Maximum PEP carboxylation rate	4.6.11	$\frac{\mu\mathrm{mol}}{\mathrm{m^2\,s}}$
T_L	Leaf temperature	4.6.25	K
T_{sky}	Effective sky temperature	4.6.17	K
T_{surr}	Surroundings temperature	4.6.16	K
Γ	CO_2 compensation point with dark respiration	4.1.18, 4.5.15	$\frac{\mu\mathrm{mol}}{\mathrm{mol}}$
Γ^*	CO_2 compensation point without no dark respiration	4.6.2	$\frac{\mu\mathrm{mol}}{\mathrm{mol}}$
Γ_s	Bundle sheath compensation point	4.5.16	$\frac{\mu\mathrm{mol}}{\mathrm{mol}}$
δ_{bl}	Leaf boundary layer thickness	4.6.20	m
ν	Air kinematic viscosity	4.6.22	$\frac{\mathrm{m^2}}{\mathrm{s}}$

Table B.2: Derived quantities (continued)

Symbol	Variable	Unit
a	Area per plant	m^2
C_A	Atmospheric CO_2 concentration	$\frac{\mu mol}{mol}$
d_L	Leaf thickness	m
f_{PAR}	Photosynthetically active radiation fraction	-
J_v	Transpiration rate	$\frac{mol}{m^2\,s}$
k	Extinction coefficient	-
LAI	Leaf area index	-
O_A	Atmospheric O_2 concentration	$\frac{mmol}{mol}$
P	Atmospheric pressure	Pa
t	Time	s
T_a	Air temperature	K
VPD	Vapour pressure deficit	kPa
VPD_a	Actual vapour pressure	kPa
VPD_s	Saturated vapour pressure	kPa
Δ	Vapour pressure slope	$\frac{hPa}{K}$
ϵ_{LU}	Light use efficiency	$\frac{g}{J}$
Ψ_c	Collar water potential	Pa

Table B.3: OpenSimRoot derived quantities and model inputs

Symbol	Variable	*Value*	Unit	Reference
a_{IR}	Infrared absorptance	0.96	-	[141]
D_{g_s}	g_s deactivation energy	264600	$\frac{\mathrm{J}}{\mathrm{mol}}$	[216]
$D_{J_{max}}$	J_{max} deactivation energy	220000	$\frac{\mathrm{J}}{\mathrm{mol}}$	[201]
$D_{V_{pmax}}$	V_{pmax} deactivation energy	214500	$\frac{\mathrm{J}}{\mathrm{mol}}$	[216]
E_{Γ^*}	Γ^* activation energy	23400	$\frac{\mathrm{J}}{\mathrm{mol}}$	[201]
E_{g_s}	g_s activation energy	116700	$\frac{\mathrm{J}}{\mathrm{mol}}$	[216]
$E_{J_{max}}$	J_{max} activation energy	37000	$\frac{\mathrm{J}}{\mathrm{mol}}$	[201]
E_{K_C}	K_C activation energy	35600	$\frac{\mathrm{J}}{\mathrm{mol}}$	[216]
E_{K_O}	K_O activation energy	15100	$\frac{\mathrm{J}}{\mathrm{mol}}$	[216]
E_{K_P}	K_P activation energy	68100	$\frac{\mathrm{J}}{\mathrm{mol}}$	[216]
E_{R_d}	R_d activation energy	41850	$\frac{\mathrm{J}}{\mathrm{mol}}$	[216]
$E_{S_{c/o}}$	$S_{c/o}$ activation energy	27400	$\frac{\mathrm{J}}{\mathrm{mol}}$	[216]
$E_{V_{cmax}}$	V_{cmax} activation energy	53400	$\frac{\mathrm{J}}{\mathrm{mol}}$	[216]
$E_{V_{pmax}}$	V_{pmax} activation energy	37000	$\frac{\mathrm{J}}{\mathrm{mol}}$	[216]
f	Spectral light quality factor	0.15	-	[201]
g_{w0}	Residual conductance	0.017	$\frac{\mathrm{mol}}{\mathrm{m}^2\,\mathrm{s}}$	[131]
g_{s25}	Reference bundle sheath conductance	0.00287	$\frac{\mathrm{mol}}{\mathrm{m}^2\,\mathrm{s}}$	[216]
J_{max25}	Ref maximum potential electron transport rate	299.6	$\frac{\mathrm{\mu mol}}{\mathrm{m}^2\,\mathrm{s}}$	[131]
K_{C25}	Reference CO2 Michaelis constant	485	$\frac{\mathrm{\mu mol}}{\mathrm{mol}}$	[39]
K_{O25}	Reference O_2 Michaelis constant	146	$\frac{\mathrm{mmol}}{\mathrm{mol}}$	[39]
K_{P25}	Reference PEP Michaelis constant	40	$\frac{\mathrm{\mu mol}}{\mathrm{mol}}$	[216]
m	Ball-Berry-Leuning slope	4.53	-	[131]

Table B.4: Constants (continued). Note that references are not provided for constants related to physics.

Symbol	Variable	*Value*	Unit	Reference
R	Universal gas constant	8.31446	$\frac{\mathrm{m^2\,kg}}{\mathrm{s^2\,K\,mol}}$	-
R_{d25}	Reference dark respiration	1.95	$\frac{\mu\mathrm{mol}}{\mathrm{m^2\,s}}$	[216]
r_m	Mesophyll dark respiration fraction	0.5	-	[201]
$S_{c/o25}$	Reference rubisco specificity	2862	-	[216]
$S_{J_{max}}$	J_{max} entropy factor	710	$\frac{\mathrm{J}}{\mathrm{K\,mol}}$	[201]
S_{g_s}	g_s entropy term	860	$\frac{\mathrm{J}}{\mathrm{K\,mol}}$	[216]
$S_{V_{pmax}}$	V_{pmax} entropy term	663	$\frac{\mathrm{J}}{\mathrm{K\,mol}}$	[216]
VPD_{ref}	Reference vapour pressure deficit	10	kPa	
V_{cmax25}	Reference maximum rubisco carboxylation rate	49	$\frac{\mu\mathrm{mol}}{\mathrm{m^2\,s}}$	[216]
V_{pmax25}	Maximum PEP carboxylation rate	119.2	$\frac{\mu\mathrm{mol}}{\mathrm{m^2\,s}}$	[216]
V_{pr}	PEP regeneration rate	80	$\frac{\mu\mathrm{mol}}{\mathrm{m^2\,s}}$	[201]
T_{ref}	Reference temperature	298.15	K	-
x	Electron transport rate partitioning factor	0.4	-	[201]
α	Absorptance	0.85	-	[201]
β	Conversion factor	2.1	$\frac{\mu\mathrm{mol}}{\mathrm{J}}$	-
Γ^*_{25}	Reference CO_2 compensation point without dark respiration	38.6	$\frac{\mu\mathrm{mol}}{\mathrm{mol}}$	[201]
ϵ_A	Carbon assimilation energy	$4.79 \cdot 10^5$	$\frac{\mathrm{J}}{\mathrm{mol}}$	[141]
θ	Empirical curvature factor	0.7	-	[201]
σ	Stefan-Boltzmann constant	$5.67 \cdot 10^{-8}$	$\frac{\mathrm{W}}{\mathrm{m^2\,K^4}}$	-

Table B.5: Constants. Note that references are not provided for constants related to physics.

B.2 Emulation

Phene	Minimum	Maximum	References
Seminal root number	0	11	[27, 82, 112]
Nodal root number	1	50	[112]
Total axial root number	17.8	105.3	[29]
Brace root whorls	0	2	[112]
Primary root branching	0.7cm^{-1}	160cm^{-1}	[112, 88]
Seminal root branching	2.5cm^{-1}	13cm^{-1}	[219, 220]
Nodal root branching	0cm^{-1}	41cm^{-1}	[154]
Seminal angle from horizontal	22	90	[112]
Nodal angle from horizontal	-5	90	[112, 193]
Brace angle from horizontal	0	90	[193]
Aerenchyma relative crossectional area	0	37.8	[112]
Lateral root length	2.2cm	17.3cm	[85]

Table B.6: Table with references

Input	Unit
shootDryWeight	g
rootDryWeight	g
"rootLength"	cm
rootLength;majorAxes	cm
rootLength;laterals	cm
rootSurfaceArea	cm^2
leafArea	cm^2
plantRespiration	g
rootCarbonCosts	g
"D90"	cm
nitrate plantNutrientUptake	µmol
nitrate UptakeEfficiencyPlant	$\frac{\mu mol}{cm^2.day}$
nitrate rootNutrientUptake;majorAxes	µmol
nitrate rootNutrientUptake;laterals	µmol
phosphorus plantNutrientUptake	µmol
phosphorus UptakeEfficiencyPlant	$\frac{\mu mol}{cm^2.day}$
phosphorus rootNutrientUptake;majorAxes	µmol
phosphorus rootNutrientUptake;laterals	µmol
potassium plantNutrientUptake	µmol
potassium UptakeEfficiencyPlant	$\frac{\mu mol}{cm^2.day}$
potassium rootNutrientUptake;majorAxes	µmol
potassium rootNutrientUptake;laterals	µmol
nitrate nutrientStressFactor	-
phosphorus nutrientStressFactor	-
potassium nutrientStressFactor	-

Table B.7: OpenSimRoot outputs at 20 days used to predict the shoot dry weight at 42 days.

Input	Unit
shootDryWeight	g
rootDryWeight	g
"rootLength"	cm
rootLength;majorAxes	cm
rootLength;laterals	cm
leafArea	cm^2
plantRespiration g	
rootCarbonCosts	g
nitrate UptakeEfficiencyPlant	$\frac{\mu mol}{cm^2.day}$
nitrate rootNutrientUptake;majorAxes	µmol
nitrate rootNutrientUptake;laterals	µmol
phosphorus plantNutrientUptake	µmol
phosphorus rootNutrientUptake;laterals	µmol
potassium plantNutrientUptake	µmol
potassium UptakeEfficiencyPlant	$\frac{\mu mol}{cm^2.day}$
nitrate nutrientStressFactor	-
potassium nutrientStressFactor	-

Table B.8: BIC-optimised subset of OpenSimRoot outputs at 20 days used to predict the shoot dry weight at 42 days.

Term	Wave 1	Wave 1.5	Wave 2 only	Wave 2 all data
x1	$3.7350 \cdot 10^{-25}$	$1.2913 \cdot 10^{-74}$	0.029528	$1.8975 \cdot 10^{-100}$
x2	$4.5452 \cdot 10^{-13}$	$7.1547 \cdot 10^{-14}$	$2.6702 \cdot 10^{-12}$	$9.3482 \cdot 10^{-17}$
x3	$1.2121 \cdot 10^{-3}$	$8.7050 \cdot 10^{-6}$	–	$2.3321 \cdot 10^{-7}$
x4	0.32877	0.16531	0.60135	–
x5	0.48031	0.22635	–	–
x8	–	–	$2.0087 \cdot 10^{-9}$	$2.9640 \cdot 10^{-5}$
x9	–	–	$1.7317 \cdot 10^{-3}$	$2.6811 \cdot 10^{-4}$
x10	$5.2573 \cdot 10^{-7}$	$2.3779 \cdot 10^{-33}$	–	$1.0817 \cdot 10^{-44}$
x12	–	–	0.013537	–
x13	–	–	$5.2387 \cdot 10^{-5}$	$1.9257 \cdot 10^{-3}$
x15	–	–	$1.9527 \cdot 10^{-6}$	$4.7558 \cdot 10^{-6}$
x16	0.018049	0.084080	–	–
x17	$4.7496 \cdot 10^{-12}$	$1.4411 \cdot 10^{-33}$	–	$2.2315 \cdot 10^{-36}$
x1x1	$3.6433 \cdot 10^{-6}$	$6.7771 \cdot 10^{-13}$	–	$1.5710 \cdot 10^{-17}$
x2x2	0.019609	$1.1537 \cdot 10^{-6}$	$1.3983 \cdot 10^{-12}$	$1.4603 \cdot 10^{-11}$
x3x3	$1.5661 \cdot 10^{-3}$	$9.3582 \cdot 10^{-5}$	–	$2.4155 \cdot 10^{-5}$
x4x4	$6.1003 \cdot 10^{-15}$	$5.8763 \cdot 10^{-28}$	$3.2552 \cdot 10^{-6}$	$1.1494 \cdot 10^{-44}$
x5x5	$6.3773 \cdot 10^{-3}$	$3.1894 \cdot 10^{-5}$	0.016446	$1.0163 \cdot 10^{-6}$
x8x8	–	–	$2.0267 \cdot 10^{-4}$	$1.4596 \cdot 10^{-3}$
x10x10	$1.0726 \cdot 10^{-5}$	$3.8922 \cdot 10^{-16}$	$8.7109 \cdot 10^{-3}$	$5.0624 \cdot 10^{-25}$
x15x15	–	–	$3.5418 \cdot 10^{-6}$	–
x17x17	$5.6738 \cdot 10^{-7}$	$2.1391 \cdot 10^{-22}$	–	$6.8572 \cdot 10^{-22}$

Table B.9: P values for linear and quadratic regression terms. Only terms selected by optimising for the Bayesian information criterion are shown.

Appendix C

Numerical Optimisations

C.1 Numerical optimisations

To calculate leave-one-out diagnostics, we take one datapoint out of the training set and train a Gaussian process on this reduced training set. This involves inverting the covariance matrix. Since the covariance matrix corresponding to this reduced training set differs from the original covariance matrix by just one row and one column, we can use the fact that we already have inverted the original covariance matrix to speed up calculations.

Let A be an n by n matrix, U an n by k matrix, C a k by k matrix and V a k by n matrix. Then the Woodbury matrix identity is

$$(A - UCV)^{-1} = A^{-1} - A^{-1}U\left(C^{-1} + VA^{-1}U\right)^{-1}VA^{-1}. \tag{C.1.1}$$

By setting

$$U = \begin{pmatrix} 0 & 1 \\ a_{21} & 0 \\ a_{31} & 0 \\ \vdots & \\ a_{n1} & 0 \end{pmatrix}, \tag{C.1.2}$$

$$V = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & a_{12} & a_{13} & & a_{1n} \end{pmatrix}, \tag{C.1.3}$$

$$C = -\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \tag{C.1.4}$$

we get

$$(A+UCV)^{-1} = \left(A - \begin{pmatrix} 0 & a_{12} & \dots & a_{1n} \\ a_{21} & 0 & & \vdots \\ \vdots & & \ddots & \\ a_{n1} & \dots & & 0 \end{pmatrix} \right)^{-1} \tag{C.1.5}$$

$$= \begin{pmatrix} a_{11} & 0 & \dots & 0 \\ 0 & a_{22} & a_{23} & \vdots \\ \vdots & a_{32} & \ddots & \\ 0 & \dots & & a_{nn} \end{pmatrix}^{-1} \tag{C.1.6}$$

$$= \begin{pmatrix} \frac{1}{a_{11}} & 0 \\ 0 & B^{-1} \end{pmatrix}, \tag{C.1.7}$$

where B is equal to A without the first row and first column. So if we have already calculated A^{-1}, we can get the inverse of a reduced matrix by calculating

$$A^{-1} - A^{-1}U\left(C^{-1} + VA^{-1}U\right)^{-1}VA^{-1}, \tag{C.1.8}$$

for U, C and V defined as above. Since C is a 2 by 2 matrix, $\left(C^{-1} + VA^{-1}U\right)^{-1}$ is very easy to calculate. Note that if we want to remove columns and rows other than the first, we first swap the rows and columns and swap them back after calculating the inverse.

To do the reverse, that is, add a single datapoint $\vec{z}$, assuming we already have calculated the inverse of the correlation matrix A, we define

$$A' = \begin{pmatrix} A & 0 \\ 0 & cov(\vec{z},\vec{z}) \end{pmatrix}, \tag{C.1.9}$$

$$U = \begin{pmatrix} cov(\vec{x}_1,\vec{z}) & 0 \\ cov(\vec{x}_2,\vec{z},0) & 0 \\ \vdots & \\ cov(\vec{x}_n,\vec{z}) & 0 \\ 0 & 1 \end{pmatrix}, \tag{C.1.10}$$

$$V = \begin{pmatrix} 0 & 0 & \dots & 0 & 1 \\ cov(\vec{x}_1,\vec{z}) & cov(\vec{x}_2,\vec{z}) & & cov(\vec{x}_n,\vec{z}) & 0 \end{pmatrix}, \tag{C.1.11}$$

$$C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \tag{C.1.12}$$

so that

$$A' + UCV = \begin{pmatrix} A & 0 \\ 0 & cov(\vec{z},\vec{z}) \end{pmatrix} + \begin{pmatrix} 0 & \dots & & cov(\vec{x}_1,\vec{z}) \\ \vdots & \ddots & & \vdots \\ & & 0 & cov(\vec{x}_n,\vec{z}) \\ cov(\vec{x}_1,\vec{z}) & \dots & cov(\vec{x}_n,\vec{z}) & 0 \end{pmatrix} \tag{C.1.13}$$

$$= \begin{pmatrix} a_{11} & \dots & a_{1n} & cov(\vec{x}_1, \vec{z}) \\ \vdots & \ddots & & \vdots \\ a_{n1} & & a_{nn} & cov(\vec{x}_n, \vec{z}) \\ cov(\vec{x}_1, \vec{z}) & \dots & cov(\vec{x}_n, \vec{z}) & cov(\vec{z}, \vec{z}) \end{pmatrix}, \tag{C.1.14}$$

which is the covariance matrix corresponding to our training set with a single datapoint added to it.

Appendix D

OpenSimRoot Guide

D.1 Foreword

This document is by no means complete and very much a work in progress. It will, at some future time, include descriptions of all the core modules of OSR for which templates are available. These descriptions will list the dependencies, if any, the input parameters needed and the various options a module has. Based on user needs and feedback, new sections will be added and existing ones will be improved. Please contact the author at ernst.schafer@nottingham.ac.uk with any requests or queries.

D.2 Introduction

It is obvious that a plant can increase its uptake of nutrients and water by growing a more expansive root system. However, the costs associated with growing and maintaining a root system mean that plants can not simply grow as many roots as they would like. The size of the root system is constrained by, among other things, the amount of available resources. This means that plants have to weigh the costs of their root systems against the benefits they provide. OpenSimRoot aims to model the three-dimensional structure of the root system, the water and nutrient uptake by the roots, the water and nutrient flow in the soil and, perhaps most importantly, the carbon costs associated with growing and maintaning a root system and taking up the nutrients needed by the plant [156]. In this way, OpenSimRoot is able to evaluate the fitness of different root architectures.

For example, using OpenSimRoot, Postma et al. [154] found that the optimal lateral root branching density depends on the availability of nitrogen and phosphorus. Inter-root competition for mobile resources such as nitrogen lead to diminishing returns on investment in lateral roots. Since the amount of carbon a plant can allocate is fixed by the shoot, higher lateral root branching density implies that the laterals are shorter, thus decreasing the efficiency. In contrast, for highly immobile nutrients such as phosphorus, there is far less competition between roots so a higher lateral root density is the more efficient architecture. Field trials show a median lateral branching density between these two extremes, suggesting that plants are finding an optimum between them. This example highlights the power of simulation in not only explaning certain features seen in field trials but also generating new questions for experimentalists.

D.3 The Structure of OpenSimRoot

OpenSimRoot is written in C++. It consists of different modules and allows the user to choose which modules to include in the simulation (though the dependency between modules imposes some restriction on the possible choices). This allows the user to study only the aspects of interest without the computational effort of running every module. The basis for OpenSimRoot is what is called the extensible tree structure. This captures not only the topology of the root system, but also provides a way to link properties to roots. Before we explain this in more detail, we will first explain in broad strokes how the engine of OpenSimRoot works. OpenSimRoot does not currently have a graphical user interface and must be run through the command line (or terminal if you are a linux or mac user).

D.3.1 The Engine

Let us first fix some definitions:

- **Minimodel.** A minimodel is, simply put, a state variable, which often depends on space and/or time. Minimodels derive from **SimulaBase**. The value of a minimodel can be requested through the application programming interface (API), which is described in more detail in section D.7.1.
- **Plugin.** A plugin is a class that computes the values of a minimodel.

Plugins derive from **DerivativeBase**. Plugins obtain information through the API and use this to compute new values.

- **Module.** We will refer a set of classes that, together, simulate a certain aspect of reality, as a module. This will involve at least one plugin and one minimodel.

As mentioned above, the information of the model is contained in the minimodels while the plugins do the calculations necessary for updating them. Each plugin is known by an unique name that the user refers to in the XML input files (see section D.5), to specify what plugin (function) OpenSimRoot should use to calculate the value of a minimodel. The **minimodels** (derive from **SimulaBase**) can be thought of as the packaging, they determine what sort of information is contained in them and what actions can be taken, while the **plugins** (derive from **DerivativeBase**) are the contents, they determine how the information inside a minimodel is calculated. This allows the user to run a simulation using different plugins for the same state variable and compare results. The values of minimodels are only calculated when requested through the API. The simulation is advanced forward in time by virtue of the export modules which start requesting the data needed for output when they are initiated.

All the minimodels derive from a class of the form **SimulaX** and these derive from **SimulaBase** (except **SimulaBase** itself of course). The **SimulaBase** class implements all general methods used to send and request information, making up the API. The derived classes each implement some methods and/or members needed when using that object type. The inheritance of the **SimulaX** classes is summarized in figure 1.

For a brief explanation of each of these classes, see section D.5. All the minimodels are linked together in what is called the extensible tree structure (ETS). This is a structure that both reflects the physical structure of the root system as well as the connections between different properties. In the most basic terms, it is a set of objects, arranged in a hierarchy with pointers to certain other objects. Each object has one *parent* and might have one or more *children*. Objects can be added and removed dynamically as needed. The XML input files also follow this structure.

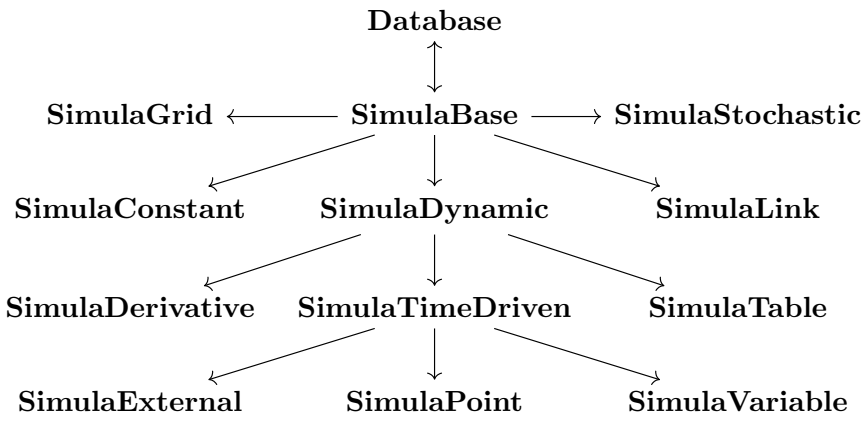


Figure D.1: Inheritance of the **SimulaX** classes. The arrow between **SimulaBase** and **Database** signifies that **Database** is privately owned by **SimulaBase**. The other arrows indicate inheritance of classes.

D.3.2 The Modules

In this section we will briefly explain the workings of some important modules in OpenSimRoot.

In OpenSimRoot each root is simulated as a number of vertices connected by edges. One of these vertices has time-dependent coordinates, it is called the growthpoint. The speed of the growthpoint is defined by the base growth rate specified in the XML input file and some correction factors that might relate to various stresses and conditions. The direction in which the growth point moves is determined according to some rules relating to gravitropism, the emergence angle of roots and a stochastic contribution. The other vertices have static locations and are placed in the path of the growthpoint as it moves. The root length is the distance the growthpoint travelled, not the sum of the distances between the vertices.

New roots are created according to branching rules which specify the distance or time between subsequent branchings. Branches emerge from what are called xylem poles, the number of xylem poles determines the radial angles at which new branches can emerge. The XML input file specifies both the axial branching angle as well as the types of roots that can branch from a certain root class. Each root class has their own parameters, such as growth rates, branching rates, etc.

OpenSimRoot contains a simple, abstract shoot model in which the shoot is represented by a number of variables. A simple photosynthesis model determines the rate of carbon production based on the leaf area and the carbon fixation rate. The carbon requirements are based on the growth and respiration rates of the roots, costs associated to root exudates and nitrate uptake and the requirements of the shoot. If the amount of produced carbon is greater than the amount required, leftover carbon is stored in a labile pool for later use. If the amount of produced carbon is smaller than the amount required, root growth rates decline.

The hydrology module in OpenSimRoot consists of three models that are linked together. One is a simplified implementation of the SWMS model in C++ which simulates water transport through the soil by numerically solving the Richards' equation [179]. The Richards' equation is:

$$\frac{\partial \theta}{\partial t} = \nabla \left[K(\theta) \nabla (h(\theta) + z) \right] - S$$

Here θ is the volumetric water content, t is time, $K(\theta)$ is the hydraulic conductivity tensor, $h(\theta)$ is the matrix head, z is the elevation above some reference point and S is a sink term that represents the water uptake by roots. Evapotranspiration, which is a term that includes the evaporation of water from the soil and transpiration by the plants, is simulated by the Penman-Monteith equation [3, 134, 133, 147]. The transport of water through the xylem is simulated by the hydraulic network model [4, 45].

The transport of nutrients in the soil is simulated with convection-diffusion equations for which there are currently two implementations in OpenSimRoot. One is the one-dimensional Barber-Cushman model that is used to simulate depletion zones around root segments [89]. The second is an implementation of the solute model in SWMS3D that couples to the water transport model [179]. The uptake of nutrients by the root system is modelled with Michaelis-Menten kinetics. The nutrient uptake rate of a root segment, I_n, is equal to

$$I_n = \begin{cases} \frac{I_{max}(C - C_{min})}{K_m + C - C_{min}} & \text{if} \quad C \geq C_{min} \\ 0 & \text{if} \quad C < C_{min} \end{cases}$$

Here I_{max} is the maximal uptake rate of the root segment, C is the nutrient concentration at the root surface, C_{min} is the minimal nutrient concentration at which the root segment can take up nutrients and K_m is the concentration at which $I = \frac{I_{max}}{2}$. If the amount of nutrients taken up is smaller than what is optimal, the plant undergoes a stress response. This can impact growth rates, photosynthesis rates and respiration rates. Nutrients and water are, in the current version, distributed instantly and uniformly around the plant, so every organ experiences the same stress. Mineralisation is modelled by the Yang-Janssen model [215].

D.4 Downloading and Running OpenSimRoot

We will describe how to download, install and run OpenSimRoot. This section is aimed at developers, there will be a standalone executable at some point in the future. The terminal commands in this section were run on a Linux machine but should be the same on Mac and Windows systems, unless explicitly stated otherwise. Example input files can be found in the OSR online repository. In the next section we will describe the structure input files should have. First of all, if you do not know what Git is, it's advisable to check out a short introduction here.

We will make a local copy of the OpenSimRoot repository (this is called *cloning*) and *build* it (we create an executable that we can run). First create an account on gitlab.com and make sure you have access to the OpenSimRoot repository. Then check if Git is installed. Do this by entering: `git --version`. Your output should be of the form: `git version 1.8.3.1`. If Git is not installed, see: `https://git-scm.com/book/en/v2/Getting-Started-Installing-Git`

Now we will add the user name and email. We do this by entering the commands:

```
1 git config --global user.name USERNAME
2 git config --global user.email EMAIL ADDRESS
```

You can check if they were entered correctly by entering:
`git config --global --list`. Now copy the HTTPS url of the online repository from the OSR Gitlab page. It can be found on the main page of the repository, there is a text box near the top of the page with a drop-down menu

beside it which is set to SSH by default. The SSH address is `git@gitlab.com:rootmodels/OpenSimRoot.git` at the time of writing. Change SSH to HTTPS and copy the link, which at the time of writing is `https://username@gitlab.com/rootmodels/OpenSimRoot.git`. `cd` to the directory that you want your local repository to end up in, `home/git` for example, and enter `git clone` followed by the HTTPS url. It will probably look like:

```
git clone https://username@gitlab.com/rootmodels/OpenSimRoot.git
remote: Counting objects: 1345, done.
remote: Compressing objects: 100% (418/418), done.
remote: Total 1345 (delta 863), reused 1328 (delta 855)
Receiving objects: 100% (1345/1345), 2.00 MiB | 2.54 MiB/s, done
    .
Resolving deltas: 100% (863/863), done.
Checking out files: 100% (310/310), done.
```

Now there should be a new directory called OpenSimRoot with all the files in it. Next we will set up our remote repository. We do this by changing directory to the OpenSimRoot directory we just created and entering the command: `git remote add origin git@gitlab.com:rootmodels/OpenSimRoot.git`

This tells git to add a remote, with the name *origin* and with as source the SSH address of the OpenSimRoot repository. You can check the remotes you added with their names with: `git remote -v`. Now whenever we want to get the latest version we make sure we are in the OpenSimRoot directory and enter: `git pull origin`. This tells git to `fetch` and then `merge`. The `fetch` command copies the latest changes from the repository and `merge` then merges these changes with the local copy of the repository. In order to avoid conflicts it is a good idea to `pull` before any local changes are made.

For more on remotes, see this webpage. For the difference between fetch and pull see this page.

```
[pmxeds@kazbek OpenSimRoot]$ git remote add origin git@gitlab.
    com:rootmodels/OpenSimRoot.git
[pmxeds@kazbek OpenSimRoot]$ git fetch origin
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (3/3), done.
```

```
remote: Total 3 (delta 2), reused 0 (delta 0)
Unpacking objects: 100% (3/3), done.
From gitlab.com:rootmodels/OpenSimRoot
 * [new branch]      master     -> origin/master
 * [new branch]      osx-redirect -> origin/osx-redirect
```

When you tell git to pull you might get the following error message:

```
[pmxeds@kazbek OpenSimRoot]$ git pull origin
You asked to pull from the remote 'origin', but did not specify
a branch. Because this is not the default configured remote
for your current branch, you must specify a branch on the
    command line.
```

This means you have not specified a default branch for your current branch (the default branch is master). You can do this by entering:
`git branch --set-upstream-to origin/master`. This will set the master branch of the remote repository (assuming you called it origin) as the default branch for your current branch (which should be your master branch). Now you can use `git push` and `git pull` without specifying the branch. To see what has been changed, use: `git diff master@{1} master` (substituting a higher integer for 1 allows you to look further back in time). Now you should get:

```
[pmxeds@kazbek OpenSimRoot]$ git pull origin
Already up-to-date.
```

We will build OpenSimRoot with gcc, which is a compiler. To see if gcc is installed, enter the commands stated below and see if the output is similar.

```
[pmxeds@kazbek ~]$ whereis gcc
gcc: /usr/bin/gcc /usr/lib/gcc /usr/libexec/gcc /usr/share/man/
    man1/gcc.1.gz
[pmxeds@kazbek ~]$ which gcc
/usr/bin/gcc
[pmxeds@kazbek ~]$ gcc --version
gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-4)
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.
    There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A
    PARTICULAR PURPOSE.
```

For more on gcc, see this webpage. Now change directory to OpenSimRoot/OpenSimRoot/StaticBuild by using `cd`

```
[pmxeds@kazbek ~]$ cd OpenSimRoot
[pmxeds@kazbek OpenSimRoot]$ ls
build.sh            executeBeforeCommitToTest.sh   public
      runTestsModules.sh
cleanup.sh          LICENSE                        README.md
      runTests.sh
OpenSimRoot         runTestsEngine.sh              warnings.txt
```

Now enter the command `bash build.sh` (Linux and Mac) or `cd` to `OpenSimRoot/StaticBuild` and use `make all` (Windows). Gcc will now build opensimroot. After a while you will see:
`Finished building target: OpenSimRoot`.

Execute: `OpenSimRoot/StaticBuild/OpenSimRoot -h` to check if everything is correct. You will get the output:

```
[pmxeds@kazbek StaticBuild]$ ./OpenSimRoot -h
Usage: OpenSimRoot [OPTIONS] [FILE]
OpenSimRoot simulates a model defined in FILE

  -f, --file                specify simulation file, default
    last argument
  -h, --help or /?          print this help message
  -v                        be verbose with warnings
  -q                        be quite with warnings
  -l, --list                print list of registered functions
  -V, --verify              Experimental function for verifying
     input files

Examples:
   OpenSimRoot -h           Prints this message
   OpenSimRoot runModel.xml Runs the OpenSimRoot model

Support at j.postma@fz-juelich.de
Licensed to you under the GPLv3 license.

There are no warnings.
```

```
Simulation took (hours:minutes:seconds): 0:0:0
```

Now test the engine and modules with the command `bash runTests.sh`. You should get the following output:

```
[pmxeds@kazbek OpenSimRoot]$ bash runTests.sh
Testing engine
using ../../StaticBuild/OpenSimRoot as exe
Test SimulaConstant.xml passed
Test SimulaTable.xml passed
Test SimulaGrid.xml passed
Test SimulaVariable.xml passed
Test SimulaPoint.xml passed
Test SimulaStochastic.xml passed
Done running tests, comparing results
Done
finished testing engine with error status 0
Testing modules
Testing OneSimpleStraightRoot.xml
using ../../StaticBuild/OpenSimRoot as exe
Test OneSimpleStraightRoot.xml passed
Test PartiallyPredefinedBranchedRoot.xml passed
Test Barber-Cushman.xml passed
Test SimpleCropModel.xml passed
Done running testing modules, comparing results
Done
finished testing engine with error status 0
exiting with error status 0
```

For a more comprehensive guide on some of the steps involved see this webpage.

We will describe how to run a simulation with OpenSimRoot. First it is important to know how OpenSimRoot operates. OpenSimRoot takes as input an XML file. This file contains the parameters for the simulation you want to run. This includes parameters like: The duration of the simulation, the environmental parameters, the initial root structure, the properties of the different types of roots etc. OpenSimRoot uses this data to run the simulation and then gives the output that the user has requested. This output can consist of aggregated variables, like the total root length, the amount of water/nitrogen depleted from the soil but

OpenSimRoot can also put out files that, for example, contain the entire root structure at each day. This can be viewed by ParaView. As a sidenote, it is possible to specify an entire root structure (with the appropriate time parameters) which OpenSimRoot will then grow. This can be used to compare the simulated results to a real result.

First we choose an output directory. It is strongly advised you make separate directories for OpenSimRoot, the XML files you will use as input and your output. While this requires you to enter slightly longer file paths in the terminal, your files will be much more organized. In the directory `OpenSimRoot/OpenSimRoot/InputFiles` there are some example XML files which we will try to run.

Use cd to move to the output directory, which we assume is `home/git/OpenSimRootOutputs`. Since our *OpenSimRoot* directory is `home/git/OpenSimRoot`, this would be:

```
cd ../OpenSimRootOutputs
```

We make a folder where we will output our test results with `mkdir Test` and then `cd Test`. Now we will tell OpenSimRoot to open the file *runStraightRoot.xml* by executing (don't copy the line break!):

```
../../OpenSimRoot/OpenSimRoot/StaticBuild/OpenSimRoot -f
../../OpenSimRoot/OpenSimRoot/InputFiles/runStraightRoot.xml
```

The first part is the path to the OpenSimRoot executable. `-f` tells OpenSimRoot that we want to open a file. The last part is the path to the file we want to take as input. After the simulation is complete, the *Test* directory contains (depending on what was specified) files called: *tabled_output.tab*, *warnings.txt*, some .vtu files and some .pvd files. The *tabled_output.tab* contains the values of variables like the nitrogen depletion, the dry weight of the plant and the total root length on each day (again depending on what output was specified). *warnings.txt* has an obvious meaning. The .vtu and .pvd files contain spatial information on the soil and root system and can be visualized with ParaView.

To do this, open ParaView (`paraview`) and open *VisualizationRoots.pvd* (*file→open*), then click *Apply* in the *Properties* window on the left. You will

probably not see anything because at time t=0 there is no root structure yet! Click the *Play* button in the top bar and you will see the root structure as it is growing. By default, the root structure is uniformly coloured so this only shows us the topology of the root system. To see more relevant information, in the properties window on the left, under the header *Coloring*, select something else instead of *Solid Color* to see the property you selected coloured (in the example below I chose rootClassID). By default the *Color Space* option (in the *Color Map Editor* on the right) is set to *Diverging*, some other colour map will probably suit you better. You can choose a different colour mapping and alter the sensitivity by adjusting the circles and curve in the *Mapping Data* section on the right. Double click on a circle in the bar at the bottom to choose a different colour.

It is also possible to see the depletion of nutrients in the soil, assuming OpenSimRoot was told to output this, by also opening the *fem.pvd* file. See Figure D.2 for a sample visualization where the different colours indicate different types of roots (primary, seminal, lateral).

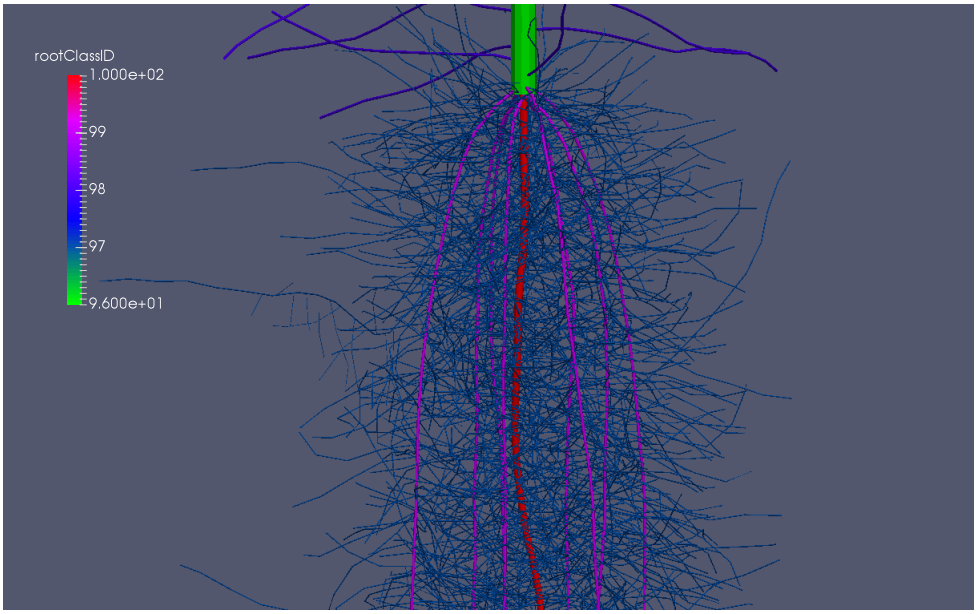


Figure D.2: A sample root system. The different colours indicate different root classes. The primary root is red, the seminal roots are purple, the hypocotyl is green, the adventitious roots are dark purple and the laterals are blue.

D.5 XML Input Files

Since OpenSimRoot aims to simulate many different aspects of root systems, a lot of parameters have to be specified. This means quite extensive input is needed. In this section we will describe the structure of input files and some of the parameters needed to run a simulation. If knowledge of the structure of the XML files is not needed, skip to section D.6 for instruction on how to create basic input files with the graphical user interface (GUI).

As mentioned before, OpenSimRoot does not have a graphical user interface and must be run through the command line (or terminal if you're a linux or mac user). When running OpenSimRoot an xml file with all the input parameters must be specified. For those unfamiliar with xml, it is a markup language similar to HTML. It can be used to store and describe data. In an xml file, information is organized in nested tags, think of them as containers that can store data or other tags. This leads to a hierarchy of tags and data which puts all minimodels in context. For example, each root class will have a tag with subtags containing the different properties of each root class.

Metadata can also be associated to each tag. In OpenSimRoot, the metadata should always at least include the name. Tags are contained in brackets: `<exampleTag>`. They are closed by either adding a `/` in front of the closing bracket or by repeating the tag with a `/` after the opening bracket. Like this:

- `<exampleTag/>`
- `<exampleTag> nested data </exampleTag>`

The outermost tag should be *SimulationModel*. The tags in the XML can be of the following types:

- **SimulaIncludeFile.** Use this tag to include another XML file. This is useful for keeping your input files (relatively) short and readable. The included file should have `<SimulationModelIncludeFile>` as outer tag.
- **SimulaDirective.** Use this to add subtags to an already existing tag. If, for example, one wants to add a property to the primary root in a file

different from the one that the primary root is defined in, one would use the tag:

```
<SimulaDirective path="rootTypeParameters/min/primaryRoot">
    <SimulaBase name = "newProperty"\>
<\SimulaDirective>
```

Here *min* is the plant type.

- **SimulaBase.** These are tags that contain other tags, such as *primaryRoot*, *branchList* or *simulationControls*.
- **SimulaConstant.** This is used for any kind of fixed parameter in the simulation. Examples are: The total time the simulation should run, the density of roots or the optimal nutrient content of the roots.
- **SimulaStochastic.** This is used for any stochastic parameter. In the current version, one can choose from the following distributions: Uniform (both integers and real numbers), normal, lognormal and Weibull.
- **SimulaTable.** This tag is used for parameters that change over time, such as the growth rate of the roots or the precipitation.
- **SimulaVariable.** Used for variables changing over time calculated using numerical integration. Examples are the actual growth rate of the roots or the water uptake rate. A plugin and an integration function need to be specified.
- **SimulaDerivative.** This tag is used for a minimodel coupled to a plugin, which needs to be specified.
- **SimulaExternal.** This tag is used for minimodels that are determined by external models, such as the water model. This external model needs to be specified.
- **SimulaLink.** This tag is used to link to other minimodels.
- **SimulaPoint.** This tag is used for points moving through space, i.e. the growthpoints.

To give a feeling for the structure of an xml input file, here is a minimal example xml file

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<?xml-stylesheet type="text/xsl" href="xml/treeview.xsl"?>
<SimulationModel name="minTestModel" date="xxx" >
    <SimulaIncludeFile fileName="minTestPlant.xml"/>
    <SimulaIncludeFile fileName="simulationControlParameters.xml
    "/>
    <SimulaIncludeFile fileName="templates/plantTemplateMinModel
    .xml"/>
    <SimulaIncludeFile fileName="plantParameters/min.xml"/>
</SimulationModel>
```

As you can see, other files are referenced here so that it is easy to enable or disable modules. The *minTestPlant.xml* file looks like:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<SimulationModelIncludeFile>
    <SimulaBase name="minTestPlant" objectGenerator="seedling">
        <SimulaConstant name="plantType" type="string">
            min
        </SimulaConstant>
        <SimulaConstant name="plantingTime"  unit="day" type="
    Time">
            0
        </SimulaConstant>
        <SimulaConstant name="plantPosition" type="Coordinate">
            0 -2 0
        </SimulaConstant>
    </SimulaBase>
</SimulationModelIncludeFile>
```

The other files referenced have similar structure. Note that this example is the absolute minimum that OpenSimRoot needs to run. If one wants to simulate geometrical aspects of the roots, the dry weight, water, nutrients or respiration one would need to add several XML files for each of these. To make OpenSimRoot more accessible for new users, sample XML files have been created. These contain the minimum number of parameters and tags needed to run certain modules. This has been done for the geometry, dry weight, water, nitrate, phosphorus and carbon modules. See the directory `OpenSimRoot/OpenSimRoot/Inputfiles`.

Understanding the structure of these XML files is vital to undestanding the inner

workings of OpenSimRoot and required knowledge for anyone who wishes to add new functionality. As seen, the outermost tag is *SimulationModel*. Within these we have **SimulaBase** tags such as *minTestPlant*, *plantTemplate*, *shootTemplate*, *dataPointTemplate* and *rootTypeParameters*. The *minTestPlant* **SimulaBase** tells OpenSimRoot to construct a seedling of type *min* at time 0 and position (0,-2,0). If more plants have to be simulated, one would add other **SimulaBase** tags with as **objectGenerator** *seedling* and specify the type, planting time and planting position of these other plants.

The "simulationControlParameters.xml" file contains the overall settings of the simulation. Examples of these are the total number of days that have to be simulated, or the values that OpenSimRoot should provide as output.

The *plantTemplateMinModel.xml* file contains the containers that OpenSimRoot will use. An example of this is the **SimulaBase** *dataPoints* where simroot will store the location of the root segments. Many of these tags are empty to start with, OpenSimRoot only requires them to exist so it can construct the object tree. Another illustration of this is the **SimulaBase** *branches* where OpenSimRoot will insert new lateral roots.

Finally, the *plantParameters/min.xml* file contains the numerical values of the parameters OpenSimRoot needs to run. In this minimal example, some of these parameters are the growth rates of the primary root and hypocotyl at different times and the gravitropism of the primary root. Naturally, when more aspects are simulated, more parameters are needed such as the distribution of nutrients in the soil, the precipitation, the transpiration rates of the plant, the leaf area etc.

It is important to mention here that generally, there is some stochasticity in certain aspects of the simulation like the branching and the growth direction of the roots. If one wants to make simulations reproducible or see the effect of a small change in the code, one has to define a random seed. This can be done using the tag

```
<SimulaConstant name="randomNumberGeneratorSeed" type="int"> 1234
</SimulaConstant>
```

which should be under the **SimulaBase** tag *SimulationControls*.

D.6 XML Generating GUI

A graphical user interface (GUI) was created that can be used to generate XML input files for OpenSimRoot. It can be downloaded from [repository]. It requires Python 3 and the TkInter and xml.etree packages, as well as some standard python packages. It is platform-independent and has a minimal, yet functional interface, see figure D.3. First select the desired templates to be included in the template selection section on the left. The GUI will automatically take dependencies between templates into account. After the right templates are selected, set the root classes. The root classes "primaryRoot" and "hypocotyl" are included by default. New root classes can be added by selecting the parent root class, typing the name of the new root class into the box below the root class selection box and clicking the "Add" button. Root classes are identified by their name, so to add a root class as a lateral to two different parent root classes, simply repeat this procedure for both parent roots. To delete a root class and all subroots from the tree, select it and press the "Delete" button. Once all the root classes are selected, the parameters can be set. Each root class has their own set of parameters, including branching parameters for each subtending root class. To set the parameters of a root class, select it in the root class selection box on the bottom left and enter them in the parameter window on the right. The general simulation settings and plant-specific parameters can be accessed by selecting "Origin" in the root class selection box. Once all parameters are set, click the "Save to XML"-button to export the simulation settings to an XML file.

The GUI works by loading the information needed from template descriptions. Every template description contains the XML snippets corresponding to that template. XML tags that contain parameters that the user might wish to alter have extra tags and information added to them that allow the GUI to recognise them. These extra tags are used to show or hide certain options depending on the state of tick boxes, display everything in the right order or add headers for readability. This structure allows developers that write new modules to add them to the GUI without having to write any code. Some functionality is still missing

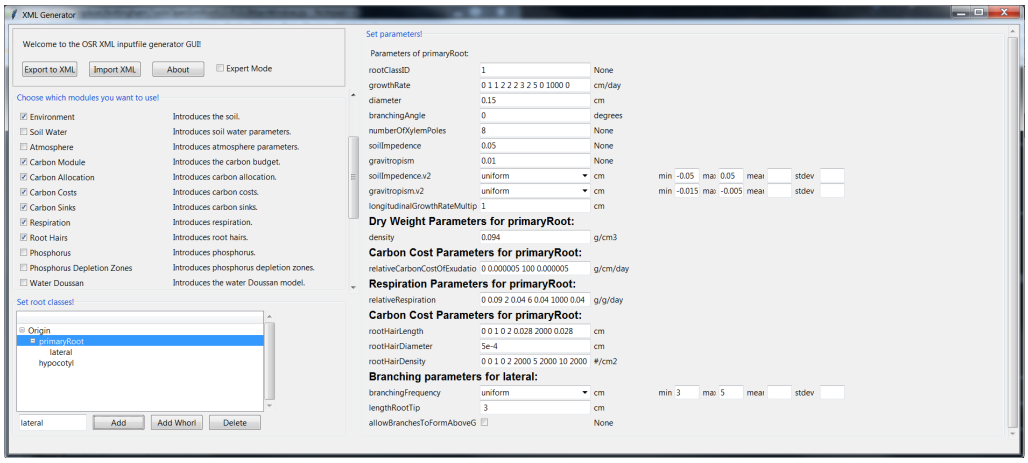


Figure D.3: The XML generation GUI as it looks on Linux operating systems.

at this time but planned to be added in the near future. This includes:

- The ability to load existing xml files into the GUI.
- The ability to copy parameters between root classes.
- The ability to process optional and alternate parameters and entries.

D.7 Writing new modules for OpenSimRoot

Writing a new module for OpenSimRoot is relatively easy, as in principle no other modules need to be changed, provided one knows C++. Being comfortable with using pointers is especially important. Each module consists of one or more minimodels and associated plugins, each of which has to have certain methods and needs to be registered into the database in a specific way. The minimal template for a new plugin is as follows:

```
#include "NewPlugin.hpp"
std::string NewPlugin::getName() const{
  return "NewPlugin";
}
NewPlugin::NewPlugin(SimulaDynamic* pSD) :
DerivativeBase(pSD){
}
void NewPlugin::calculate(const Time &t, double &d){
}
DerivativeBase * newInstantiationNewPlugin(SimulaDynamic* const
    pSD){
```

```
  return new NewPlugin(pSD);
}
class AutoRegisterRootLossInstantiationFunctions {
public:
  AutoRegisterRootLossInstantiationFunctions() {
    // register the maker with the factory
    BaseClassesMap::getDerivativeBaseClasses()["newPlugin"] =
    newInstantiationNewPlugin;
  }
  ;
};
static AutoRegisterRootLossInstantiationFunctions p5236245;
```

The header corresponding to this should look like:

```
#ifndef NEWPLUGIN_HPP_
#define NEWPLUGIN_HPP_
#include "../../engine/BaseClasses.hpp"

class NewPlugin: public DerivativeBase{
public:
  NewPlugin(SimulaDynamic* pSD);
  std::string getName() const;
protected:
  void calculate(const Time &t, double &var);
};
#endif
```

Each plugin should have a constructor with as argument a pointer to a **SimulaDynamic**, *pSD*. This is used as a starting point for navigating around the extensible tree structure. It should also have the methods **getName** and **calculate**. The **getName** method simply returns the name of the plugin. This is used to identify the right classes and to navigate the extensible tree structure. The **calculate** method will calculate and return the value of the associated minimodel. In the *simulationControlParameters.xml* file (or any other file that contains the relevant parameters) one can specify how often values are written to the output. The minimodels that record global variables will write these to the *tabled_output.dat* file as specified.

D.7.1 The API

Note: When writing a module, one invariably needs to obtain values from other modules and thus knowing how to navigate the extensible tree structure is essential. We will now explain the most common methods used. Assume that all these methods are called from a **SimulaBase**-object called *current*.

- `SimulaBase* getParent()`

  This method returns a **SimulaBase***, a pointer to the **SimulaBase** that is the parent of "current". The method is overloaded and can take a positive integer as input. This will prompt the method to go up this number of steps, so choosing **2** as input will prompt it to return the 'grandparent' instead of the parent, etc.

- `SimulaBase* getChild(const std::string& childName)`

  This method returns a **SimulaBase*** pointing to the child of "current" which is called "childName". Of course, if "current" does not have a child with that name, it will lead to an error message. This is why the following method exists.

- `SimulaBase* existingChild(const std::string& childName)`

  This method works similarly to the above one, except that if no child with the given name exists, it returns a null pointer. If one wants to get a value from an optional module, on the condition that it is being used, this method is used instead of the previous one.

- `SimulaBase* getSibling(const std::string& siblingName)`

  This method is equivalent to

  ```
  getParent()->getChild(const std::string& siblingName)
  ```

  but obviously preferable. The method

  ```
  SimulaBase* existingSibling(const std::string& siblingName)
  ```

  works just like one expects.

- `SimulaBase* getNextSibling()`

  This method returns a pointer to the next sibling in the lexicographical ordering. This is useful for iterating over all the datapoints of a root in order, as they are created in lexicographical order.

- `SimulaBase* getPreviousSibling()`

  This method returns a pointer to the previous sibling in the alphabetical ordering.

- `SimulaBase* getFirstChild()`

  This method returns a pointer to the first child of "current" in the alphabetical ordering. Useful for the start of an iteration over all children.

- `SimulaBase* getLastChild()`

  This method returns the last child in the alphabetical ordering.

- `void getAllChildren(SimulaBase::List& childrenList)`

  A **SimulaBase::List** object is a vector of **SimulaBase*** objects, so a vector of pointers. This method copies the list of all children of "current" to childrenList. This is useful when iteration over elements is needed.

- `std::string getName()`

  This method returns the name of the SimulaBase object from which it's called, in this case that would be "current".

- `std::string getPath()`

  This method returns a string that describes the path from the origin to "current". Can greatly improve the usefulness of error messages. For example, the path to the first datapoint of the primary root could be:
  `origin/examplePlant/plantPosition/primaryRoot/`
  `dataPoints/dataPoint00000`

- `SimulaBase* getPath(const std::string& path)`

  This method returns a pointer to the SimulaBase object specified by "path". If unsure if this object exists, one can use

  `SimulaBase* existingPath(const std::string& path)`

To navigate the extensible tree structure (ETS), one obviously has to know its topology. To get an overview of the structure of the ETS, turn on the *modelDump*-option. This will write the ETS at the specified times to an XML file. It can be viewed in any internet browser. One might have to delete the second line, which is shown below, on some systems.

```
<?xml-stylesheet type="text/xsl" href="XML/outlineview.xsl" ?>
```

It is advisable to not do a model dump for simulations that have run for more than a couple of days, the output file might turn out to be several hundreds of MB in size.

D.8 Acknowledgements

We would like to acknowledge the developer of OpenSimRoot, Dr. Johannes Postma, whose comments and corrections were essential for this manual. We would also like to acknowledge Dr. Nathan Mellor, discussions with whom helped clarify many of the aspects of OpenSimRoot we discussed. Finally we would like to thank Prof. Markus Owen for his extensive feedback.