Theory of Hybrid Systems Informatik 2 | RWTH AACHEN UNIVERSITY

Diese Arbeit wurde vorgelegt am
Lehr- und Forschungsgebiet Theorie der hybriden Systeme

Quantification of Uncertainties and Determination of Sensitivities for Biogeochemical Ocean Simulations

Quantifizierung von Unsicherheiten und Analyse von Sensitivitäten in biogeochemischen Ozeansimulationen

Masterarbeit
Informatik

April 2021

Vorgelegt von Presented by	Marvin Clive Birger Thelen Kruppstraße 18 52072 Aachen Matrikelnummer: 344746 marvin.thelen@rwth-aachen.de
Erstprüfer First examiner	Prof. Dr. rer. nat. Erika Ábrahám Lehr- und Forschungsgebiet: Theorie der hybriden Systeme RWTH Aachen University
Zweitprüfer Second examiner	Prof. Sebastian Krumscheid, PhD Lehrstuhl für Mathematics for Uncertainty Quantification RWTH Aachen University
Externer Betreuer External supervisor	Dr. rer. nat. Pascal Richter Lehr- und Forschungsgebiet: Theorie der hybriden Systeme RWTH Aachen University

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich diese Masterarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Die Stellen meiner Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken entnommen sind, habe ich in jedem Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht. Dasselbe gilt sinngemäß für Tabellen und Abbildungen. Diese Arbeit hat in dieser oder einer ähnlichen Form noch nicht im Rahmen einer anderen Prüfung vorgelegen.

Aachen, im April 2021

Marvin Thelen

At this point I would like to express my gratitude towards all the people who supported and accompanied me during the development of this work.

I would like to thank both my thesis advisors Dr. Pascal Richter and Sebastian Krumscheid PhD for all the support and guidance during the creation of this thesis. I deeply appreciate all the encouragement and constant revision of my work.

Furthermore, I would like to acknowledge the continuous collaboration effort by researchers at the Alfred-Wegener-Institut. Special thanks go out to Nabir Mamnun, Dr. Lars Nerger and Dr. Christoph Völker whose consultation immensely advanced the progress of this thesis. Our joint discussions and the constant adjustment of their preparatory work were always highly appreciated.

Contents

1. Introduction

Environmental sciences describe the interdisciplinary study about physical, chemical and biological conditions of the environment and their effects on organisms. This academic field has gained attention over the last few decades as many studies emerged to examine the extent of anthropogenic actions destabilizing our long-established ecological balance. Besides exhaustive exploitation of resources and the emerging problem of waste management, climate change due to greenhouse gas emissions is probably the most prominent field of research in this area.
Climate as a statistical entity arises from the assessment of several meteorological variables, for example temperature or atmospheric pressure, which are averaged over a long period of time for a given location. As such it contrasts the short-term expressiveness of weather since models based on climate statistics provide the framework to extrapolate climate conditions for years to come. Weather records exist for roughly 150 years, so in order to compensate for that relatively short amount of time, scientists use proxy evidence like ice cores or ocean sediments to reconstruct developments in paleoclimatological time frames [11]. Figure 1 shows a series of reconstructed conditions regarding temperature deviations from recently measured means. The modelled datasets suggest

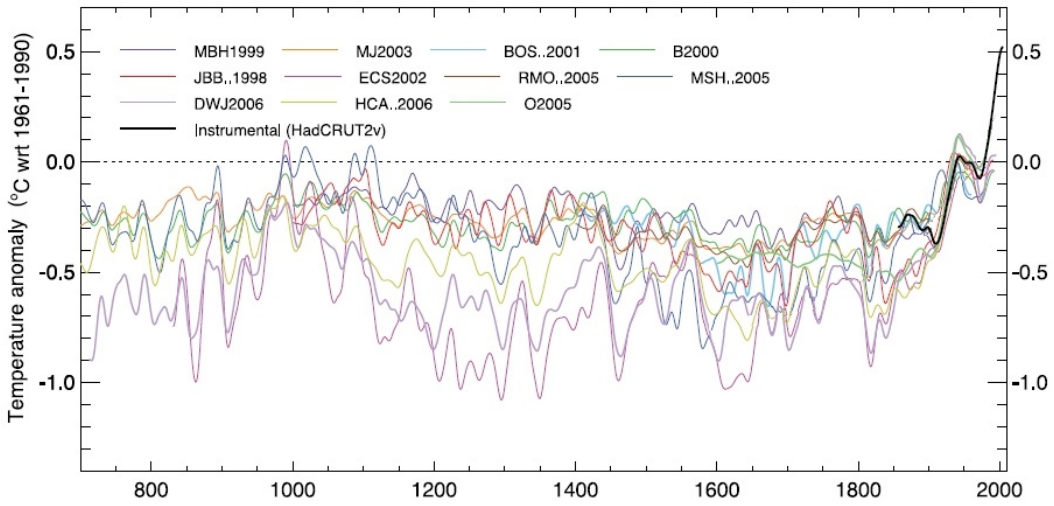


Figure 1: Reconstructed estimations of temperature anomalies [36]. Note that empirical data represented by the black line is only available starting in the second half of the 19^{th} century.

a clear upward trend in temperatures since at least one hundred years which can be verified by the empirical data acquisition of *HadCRUT* [32]. Even though one can argue on basis of the Milankovitch cycle that temperature fluctuations have been present throughout time, Crowley among others argued that recent increases cannot be justified by natural variability alone [13].
Findings of anthropogenic temperature changes is usually intertwined with unnatural excess of carbon dioxide in the atmosphere which scientists believe to be one of the main culprits causing the greenhouse effect [47]. *Scientists For Future* as one of those

groups incorporates a data visualization for global temperature change in their current logo (see Figure 2) that aims at increasing public awareness about this issue on a popularized level.

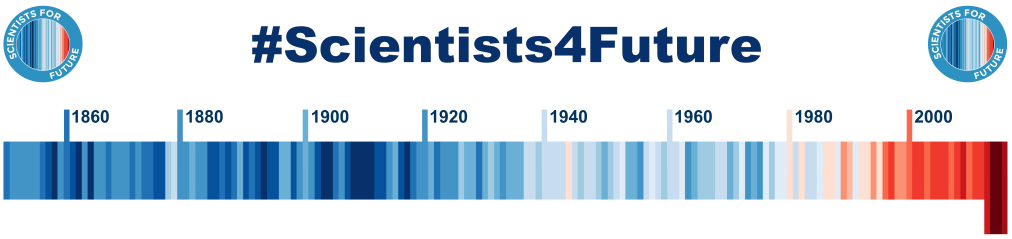


Figure 2: Banner from *Scientists for Future* [18]. It shows the group's logo and a warming strip from 1850 to 2018 that visualizes the temperature in each year compared against the average temperature of this period. The darker the colour the greater the distance to the average, where blue refers to colder and red refers to hotter temperatures than average in corresponding years. According to them the years 2015 to 2018 were the warmest years since the beginning of weather records.

In the long term the distribution of carbon in our environment is in a balanced state. We can still identify certain cycles which transfer carbon from one pool to another. With regard to the greenhouse effect, we perturb an intrinsically slow cycle of carbon bound in fossil fuels. Those compounds form from organic matter that has been compressed by pressure and heat under layers of mud for millions of years. Volcanic activity is the main driver for the counter rotation of carbon into the atmosphere, which is a relatively slow process as well under natural circumstances. By excessively burning fossil fuels, humans have greatly accelerated this cycle unilaterally and thus increased atmospheric carbon levels to 410 ppm from a pre-industrial concentration of around 270 ppm [5].
Less than half of anthropogenic emissions remain in the atmosphere which implies an absorption of carbon surplus by either terrestrial plants or the ocean [43]. Plants generally experience a boost in biomass production when more carbon becomes available as long as water and other nutrients do not limit their growth. The other player in this system is the vast ocean with its biotic and abiotic carbon pumps. Latter is often referred to as solubility pump that is shaped by a carbon dioxide exchange via the sea-air interface. A quite simplistic view on carbon intake of the ocean presumed that mere measurements of carbon concentrations at two points in time would reveal the anthropogenic proportion in this matter. It became clear, however, that there is more to it because a multitude of factors from winds to fluid dynamics and nutrient supply for primary production of algae and plankton significantly alter the carbon cycle in the ocean and thus the global cycle. As a consequence, biogeochemical models emerged in the science community to better understand and predict implications for the climate as a whole.

1.1. Related Work

This thesis resides in the context of a Biogeochemical Ocean Model. Its embedment within the model community is depicted in the next section. Afterwards, the area of Uncertainty Quantification (UQ) is explored where we discuss several methods that are further investigated in the course of this work. We later present the methodical approach of a Sensitivity Analysis (SA) that is deeply ingrained in the preceding Uncertainty Quantification.

1.1.1. Model

Global climate models in the 1950's started off as computerized attempts to create a representation of the earth's atmospheric system as it was at that time [39]. By the time scientists recognized anthropogenic actions as a driving force for climate change, models had become detailed enough and computers more powerful in order for models to gain prognostic capabilities, e.g. the Community Climate Model [34]. Despite increased computational resources, simulations are not even remotely able to render down to a molecular level. Modern projections in the family of General Circulation Models (GCMs) therefore divide the examined three dimensional space volume of the atmosphere and the ocean into cell compartments of feasible size. We distinguish between an Atmosphere GCM (AGCM), an Ocean GCM (OGCM) or their coupled AOGCM which can naturally be even more detailed, but also more complex. An example of a state-of-the art model for latter category is *GEOS-5* that can analyse climate variability on a wide range of time scales [6].

When it comes to OGCMs, ocean fluid dynamics have advanced our understandings of physical and thermodynamical processes in the ocean on the basis of the Navier-Stokes equations. Biogeochemical models are often coupled with OGCMs to form Biogeochemical Ocean GCMs (BOGCMs), but in contrast to standalone OGCMs they have no known equivalent to these techniques of numerical modelling. The chemical component of biogeochemical models are widely explained with stoichiometric relations, but things become significantly more complex once transformations among organic and inorganic compounds are considered. Marine ecosystems are thus often divided into compartments that are intertwined by transportation of compounds to one another. NPZD-type models as seen in Figure 3 originate from a version of Fasham et al. and they incorporate a rough division into compartments of nutrients, phytoplankton, zooplankton and detritus [17]. Modifications of these models integrate even more compounds like phosphorus and nitrogen. General improvements include eddy-permitting ocean circulation models like *ORCA-LIM* [27] that further advances towards real ocean conditions. More recent models like the Dynamic Green Ocean Model tend to go beyond the division seen in NPZD-type models and for example further break down the distinction between both supergroups of phytoplankton and zooplankton and result in far more parameters used in the model [7].

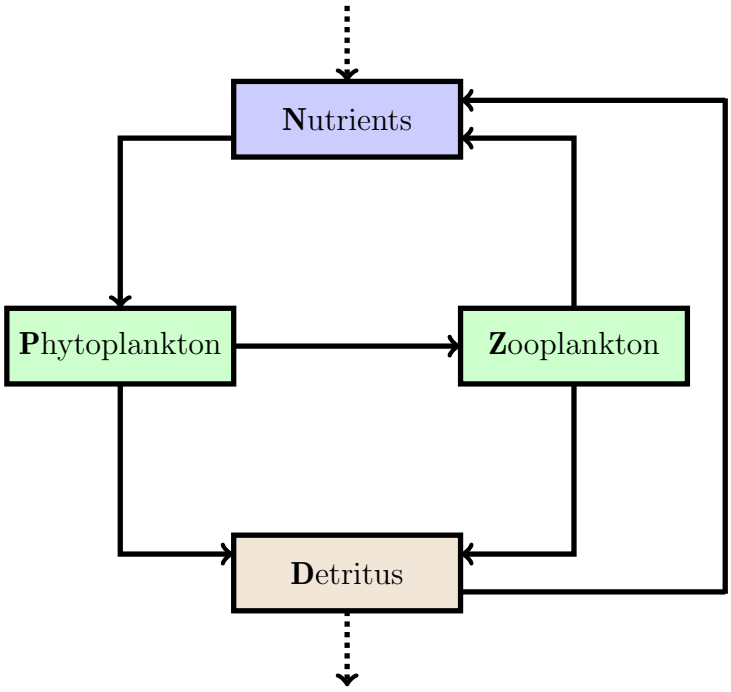


Figure 3: Layout of a classic NPZD-type model with its compartments of Nutrients, Phytoplankton, Zooplankton and Detritus. Arrows between compartments indicate the fluxes between them. The dotted arrows imply boundary interactions with the atmosphere and the benthos.

1.1.2. Uncertainty Quantification

Uncertainty Quantification (UQ) and Sensitivity Analysis (SA) often go hand in hand, where the former can be viewed as a prerequisite for SA. Uncertainty Quantification in the context of our thesis firstly deals with the characterization of uncertainties within a parameterized model, whereas the Sensitivity Analysis generally refers to an assessment of the sensitivities in the model output with respect to changes in the input values.

It is important to distinguish between different sources of uncertainty because they determine how they can be quantified. De Wit and Augenbroe categorized sources into model, numerical, specification and scenario uncertainties, where the last two can be summarized as model input uncertainty [15]. Model uncertainty is attributed to the fact that models are inherently only approximations of the real world and thus lack absolute accuracy. An example for these in biogeochemical models are unknown future greenhouse gas emissions or natural variability [21]. Numerical uncertainties arise from computational limitations. These include storing floating numbers in a computer number format which need to be rounded to some extent or discretization errors in the execution of simulations. In the context of Building Simulations, De Wit and Augenbroe argued that numerical uncertainties can be neglected. For the sake of focussing on more influential uncertainties, we will also refrain from taking model and numerical

uncertainties into account.
In order to quantify input uncertainty, Saltelli and Tarantola suggested techniques ranging from physical bounds and estimates to expert judgement and measurements [45]. Even though it is viable to rely on literature and experts for a rough quantification, most applications only incorporate their assessments in the determination of for example the choice of a suitable probability density function around model parameters. Depending on their individual context, Macdonald presented implications of different distributions for complex simulation models [28]. Such distributions set the basis for Monte Carlo techniques which are commonly used in the background of UQ and SA. Cox and Baybutt included Monte Carlo simulations in their distinction of approaches in probabilistic risk assessment, but argued that Monte Carlo techniques have the disadvantage that they are computationally expensive. They rather recommended the use of analytical techniques if applicable to the underlying model. However, their suggestions are dated back to 1981 and computational power has increased greatly and thus lead to Monte Carlo simulations as commonly preferred method [12].
Another way of quantifying uncertainties was described by Oakley and O'Hagan who employed a Bayesian approach [35]. The main idea is to observe output values and try to find input values X^* that can produce this output. Since an exact solution is not reachable in practice, an approximation is sought with an observation error kept as small as possible. A Bayesian framework hence shifts the perspective from forward propagated uncertainties to an *inverse* setting. Instead of choosing samples uniformly at random, the Bayesian method considers previous runs and hence is able to perform more efficiently. Note that the key requirement and also downside is that the underlying model function is assumed to be a smooth function in order to extract viable information from X^*.

1.1.3. Sensitivity Analysis

Sensitivity Analysis facilitates the calibration of models and the identification of critical regions for input parameters. We distinguish between local and global SA methods. Former focuses on a local range around a base point in the parameter space and thus is limited in its effectiveness if the parameter region is not known. Derivative based methods as applied by Rao and Haghighat count as local SA and employ the partial derivative $\frac{\partial Y_i}{\partial X_i}$ for an input X_i and output value Y_i [41]. Computing these derivatives algebraically, however, requires the model to be formulated in an explicit algebraic equation. Note that this is not the case for our underlying biogeochemical ocean model *REcoM2* . The one-at-a-time (OAT) method overcomes this problem by varying parameters from their base values. This is performed for every parameter consecutively while leaving all other parameters at their original value. OAT can hence be viewed as a method for the approximation to the partial derivative method. Saltelli and Annoni presented the inefficiency of an OAT method since it does not capture the interactions between input parameters and as a consequence they suggested the use of global methods [44].
For a Sensitivity Analysis of complex and non-linear models like *REcoM2* , Burhenne

considered global methods as the preferable choice since they analyse the parameter space with respect to their corresponding output of the model [8]. If the number of parameters is relatively low, the Scatter Plot Method is capable of quickly distinguishing between influential and non-influential parameters. On the basis of a Monte Carlo simulation, sample values together with their result (i.e. $\left(x^{(1)}, y^{(1)}\right); \ldots; \left(x^{(N)}, y^{(N)}\right)$) are plotted. In this setting, a uniform cloud of points for example could be an indicator of a non-influential parameter. Drawbacks are the necessity of a visual inspection of every graph and partial loss of granularity if parameters are not dominant enough to produce a visible pattern.
Campolongo et al. made an effort to present advantages of variance-based SA methods like the independence of the underlying model and their inclusion of interactions between uncertain parameters [10]. A variance-based method managed to perform well for a numerical model by Zhao and Tiede that examines only five parameters randomly sampled from a uniform distribution [64]. Even though variance-based methods have a high computational overhead, they can be used for most applications. More importantly, they allow us to perform a Sensitivity Analysis that answers two of the main questions of this thesis: Which parameters of the Biogeochemical Ocean Model are highly sensitive and its inverted counterpart question of which parameters can later be neglected?

1.2. Goal of This Thesis

In this work, we are looking at the Regulated Ecosystem Model 2 (*REcoM2*) which simulates among other output variables the net primary production of phytoplankton and the concentration of chlorophyll in the ocean. Due to its complexity, the accuracy of the model is highly susceptible to uncertainties which are expressed in its parameters. Our goal is to quantify those uncertainties and to subsequently perform an analysis of their respective sensitivity. Even though we will not directly decrease its complexity, a sensitivity analysis will hopefully improve the forecast accuracy of the model by providing insights on the sensitivity of its input parameters. Furthermore, it could potentially benefit broader climate models that incorporate results from the underlying biogeochemical ocean model.

1.3. Outline

Section 2 will cover some technical processes of the underlying biogeochemical model *REcoM2*. An introduction into the methodology of a Sensitivity Analysis in the context of Uncertainty Quantification is given in Section 3. In order to explain the principle of our Sensitivity Analysis, Section 4 demonstrates the codebase used in this thesis and also presents academic test cases for an analysis. After that we will present results of a Sensitivity Analysis that is performed on the biogeochemical model and offer insights into the sensitivity of parameters in Section 5. Finally, Section 6 will summarize our results and will give an outlook on future work that is beyond the scope of this thesis.

2. Biogeochemical Model

In the following we present the biogeochemical model *REcoM2* . Goal of this chapter is to give a rough outline of the model in Section 2.1 and explain its mode of operation in a more technical way in Section 2.2.
REcoM2 is usually not used as a standalone model and requires an Ocean General Circulation Model as physical simulation component. A coupling has been performed for instance with the model MITgcm which is also the coupling we employ for our case study [26]. In the following we describe a coupling with the ocean model FESOM which does not greatly diverge in substance, but rather in notation. This fact is of little significance since this thesis is not concerned with a thorough analysis of the model itself. Note that we will refer to a coupling only as *REcoM2* .

2.1. REcoM2 Overview

Ocean biogeochemical processes are inherently influenced by circulation and turbulent transport processes within the ocean medium. In models these are approximated through coupled OGCMs. Photosynthetically available radiation as well as fluxes at the sea-air interface are also considered in their framework. Figure 4 shows the general relationship between common model actors and mutual interactions.
We now take a closer look at the biogeochemical model *REcoM2* which will be used throughout this work. A full description is provided by Schourup-Kristensen et al. [49]. With regard to Section 1.1.1, *REcoM2* can be considered an advanced NPZD-type model because besides the basic Nutrients, Zooplankton and Detritus compartment, the phytoplankton are further divided into nanophytoplankton and diatoms. The benthos (biogeochemical processes at the seafloor) plays an important part in the nutrient cycle and is also incorporated as a compartment. An overview of those compartments and the fluxes between them is given in Figure 4.

REcoM2 considers the air-sea gas exchange of carbon in accordance to findings by the Ocean Carbon Model Intercomparison Project and Wanninkhof [37, 60]. One driving force for carbon exchange is a combination of a thermodynamic property of CO_2 and ocean circulation, the so-called physical carbon pump. The solubility of CO_2 in sea-water is greater at lower temperature and the ocean is therefore capable of taking up more CO_2 at high latitudes. As the dense water formed in these regions sinks and subsequently spreads into the interior of the ocean, a vertical gradient of dissolved inorganic carbon (DIC) is produced. The deep and CO_2-rich water masses are transported by the global ocean circulation and close the cycle by upwelling and mixing with the surface ocean, where solubility decreases and outgassing happens [38].
Even though the solubility pump is partially responsible for the vertical DIC gradient, about three quarters of that phenomenon are attributed to the biological carbon pump [59]. It is tightly coupled to the compartments of *REcoM2* and emphasizes the importance of biogeochemical ocean models for the evaluation of global carbon cycle. Photosynthetically active phytoplankton generate organic carbon mainly in the euphotic zone situated at the surface ocean and thereby decrease DIC concentrations

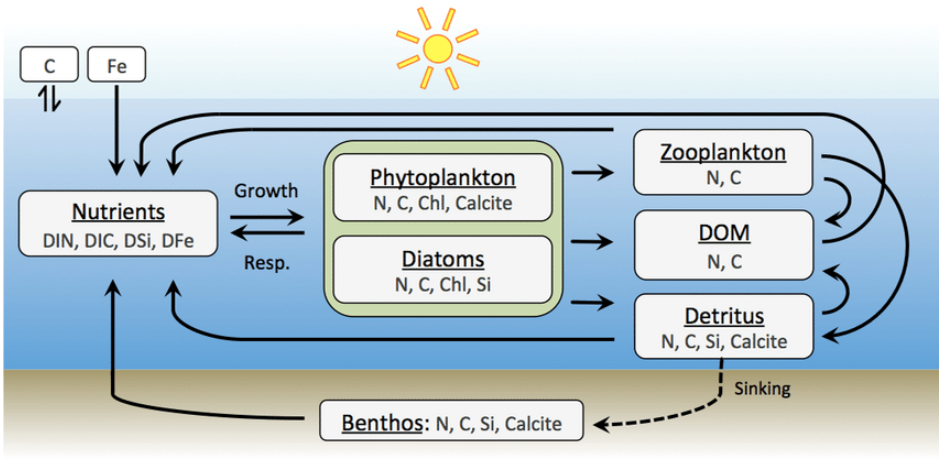


Figure 4: *REcoM2* compartments together with the pool of compounds associated with them [49]. Arrows indicate fluxes between compartments.

and the partial pressure of CO_2 (pCO_2) accordingly. The opposite effect comes from metabolic processes (respiration) in zooplankton as high trophic organisms, in the phytoplankton themselves as well as in bacteria. Since biogenic particles tend to sink in the water column rather than follow the forces of advection, respiration generally occurs in lower ocean layers than photosynthesis and thus contributes to the vertical DIC gradient. A particular subset of nanophytoplankton (coccolithophores) in our model form calcium carbonate ($CaCO_3$) which also affects the carbon cycle. Its formation increases pCO_2 and creates a countering effect to the aforementioned biological pump in terms of CO_2 concentration at the surface ocean. The sinking of particulate $CaCO_3$, the subsequent dissolution of $CaCO_3$ at depth and the transport of the remineralized material back to the surface is sometimes called the $CaCO_3$ counter pump. Detritus acts as the bridge compartment that maintains the nutrient supply back to the upper ocean layers by ocean circulation dynamics.
Observe that we have only focused on the carbon specific cycle in *REcoM2* . Besides carbon, the model also incorporates nitrogen, silicon and iron as nutrient compounds. Iron is an important factor for primary production and new supply of iron is included from aeolian dust at the sea-air interface and from input of the sediment. Nitrogen and silicon generally undergo the same cycle as carbon, from their dissolved state to organic matter as nutrients and later on mostly back to the nutrient pool through remineralization.
The reasoning for splitting phytoplankton into nanophytoplankton and diatoms becomes apparent when inspecting their respective cell structures. Nanophytoplankton form calcite shells that we discussed above, whereas diatom cells are contained within a silica cell wall. *REcoM2* thereby considers silicon and calcite exclusively for diatoms and nanophytoplankton, respectively. With regard to zooplankton nutrient uptake, the model only recognizes carbon and nitrogen as well as the factors for the grazing of both phytoplankton types.

Boundary conditions do not only apply to the atmosphere (CO_2 gas exchange and iron input), but also to the ocean floor. Detritus is split into nitrogen, organic carbon, calcite and silicate. Matter from the detritus which is not dissolved during sinking is provided to the benthos and in turn is remineralized and given back to the pool of inorganic matter.

2.2. Model Formulas

We now present more detailed information about the internal processes of *REcoM2* . This includes used variables, their mutual dependencies and specific formulas used for calculation. Note that this model is only the basis for the specific implementation we will be using and that certain workflows may differ from the actual program.
Biochemical processes of the model are embedded in the ocean medium. *REcoM2* is hence exposed to the physical ocean circulation mechanisms and biochemical processes within the water on the one hand and impacted by boundary conditions from the atmosphere and benthos on the other hand.
The benthos has been fully included in the model and especially in the nutrient cycle, whereas the atmosphere is not represented by state variables and is only responsible for iron input and CO_2 fluxes at the sea-air interface. Orr et al. provided the guidelines from which the CO_2 flux calculations are derived [37].
Biological state variables like the concentration of nutrient compounds within the water body or within the cells of primary producers are constantly affected by ocean circulation and biological processes. The goal is to calculate the change in concentration of a biological state variable S in a given volume of water. The water's velocity in x,y and z directions is given by $U = (x, y, z)$. Dead organic matter is supposed to sink with a gravitational settling velocity that increases linearly with depth. The velocity w_{det} is determined by

$$w_{\text{det}} = \frac{0.0288}{\text{day}} \cdot z + w_0 \tag{1}$$

where z is the current depth in meters and w_0 is the sinking speed at surface level found as parameter in Table 24. Then, the term $-U \cdot \nabla S$ describes the change in S due to advection. Turbulent motion is also considered by the term $\nabla \cdot (\kappa \cdot \nabla S)$, where κ represents the diffusity tensor. Biological processes that act as either *source* or *sink* are taken into account through the function SMS(S), where SMS stands for "sources minus sinks". Those functions are yet to be individually illustrated for a few state variables. With our terms for advection, turbulent motion and the SMS-function, we attain the following equation for change in concentration over time:

$$\frac{\partial S}{\partial t} = -U \cdot \nabla S + \nabla \cdot (\kappa \cdot \nabla S) + SMS(S) \tag{2}$$

2.2.1. Sources Minus Sinks

Formulas and naming conventions in this section have been adapted from [49].
We distinguish between state variables of the model (Tables 18-21) and its parameters

with fixed values (Tables 22-27). Table 18 and Table 19 depict the state variables for which we will calculate the SMS terms. Fluxes from or to the benthos are denoted in variables in Table 21 and are partially dependent on ocean or benthos state variables. Lastly, Table 20 contains general model variables that are used in several SMS functions.

In light of the sheer mass of functions that arise from the model, we only present a handful of formulas that transport the essence of *REcoM2* . If not explicitly mentioned otherwise, we refer to the tables in Appendix A.1 for an explanation of symbols in the equations. Terms that are highlighted by a green color represent sources whereas the color red depicts sinks.

Dissolved inorganic nitrogen (DIN):

$$SMS(DIN) = \underbrace{p_N \cdot f_T \cdot DOM}_{\text{DON remineralization}} - \underbrace{V^N_{\text{nano}} \cdot PhyC_{\text{nano}}}_{\text{Nitrogen assimilation}} - \underbrace{V^N_{\text{dia}} \cdot PhyC_{\text{dia}}}_{\text{Nitrogen assimilation}} \tag{3}$$

$p_N \cdot f_T$ describes the temperature dependent remineralization of dissolved organic nitrogen (DON). V^N_{nano} and V^N_{dia} are the nitrogen assimilation rates for nanophytoplankton and diatoms, respectively. Intracellular carbon concentration in nanophytoplankton and diatoms, respectively, is denoted by $PhyC_{\text{dia}}$ and $PhyC_{\text{nano}}$. DOM is an acronym for *dissolved organic matter*.
The remineralization of dissolved organic nitrogen has a temperature dependency (f_T) which is derived by an Arrhenius function with T as the local temperature:

$$f_T = \exp\left(-4500K \cdot \left(\frac{1}{T} - \frac{1}{T_{\text{ref}}}\right)\right) \tag{4}$$

T_{ref} as the reference temperature is parameterized in Table 27. Figure 5 illustrates the temperature dependency function.

Dissolved iron (DFe):

Dissolved iron is accounted for by a constant $Fe : N$ ratio which is denoted as $q^{\text{Fe:N}}$ in Table 23.

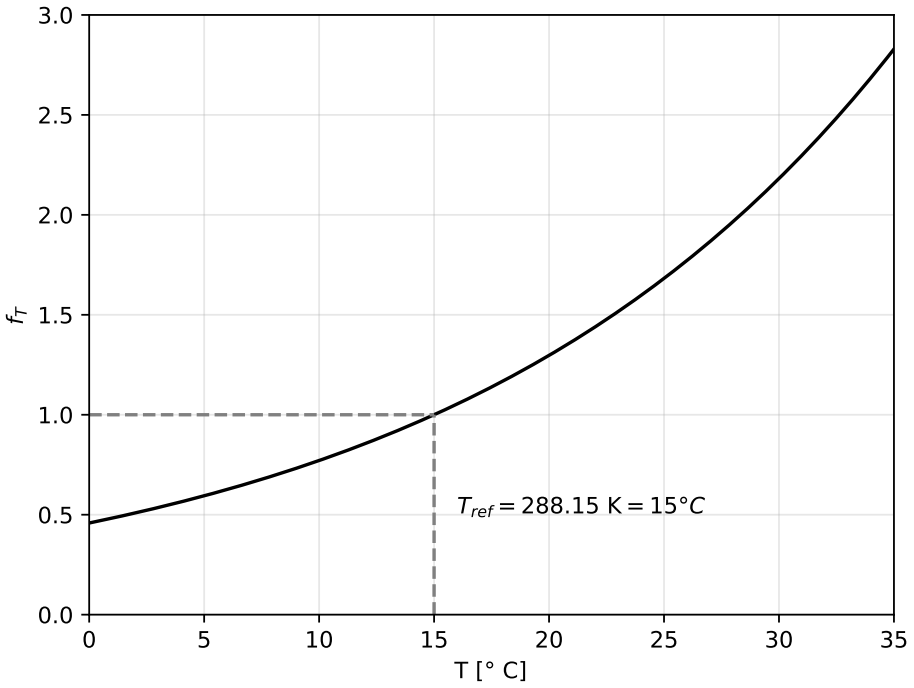


Figure 5: The Arrhenius function f_T. Note that its output is 1.0 if $T = T_{\text{ref}}$.

$$
\begin{aligned}
SMS(DFe) = {} & q^{\text{Fe:N}} \cdot \underbrace{\epsilon_{\text{N}}^{\text{phy}} \cdot \left(f_{\text{lim,nano}}^{\text{N:Cmax}} \cdot PhyN_{\text{nano}} + f_{\text{lim,dia}}^{\text{N:Cmax}} \cdot PhyN_{\text{dia}} \right)}_{\text{Phytoplankton excretion}} \\
& + q^{\text{Fe:N}} \cdot \underbrace{\rho_{\text{DetN}} \cdot f_T \cdot DetN}_{\text{Detritus remineralization}} \\
& + q^{\text{Fe:N}} \cdot \Big(\underbrace{\epsilon_{\text{N}}^{\text{zoo}} \cdot ZooN}_{\text{Zooplankton excretion}} + \underbrace{r_{\text{zoo}} \cdot ZooN}_{\text{Zooplankton respiration}} \Big) \\
& - q^{\text{Fe:N}} \cdot \underbrace{V_{\text{nano}}^{N} \cdot PhyN_{\text{nano}}}_{\text{Nanophytoplankton net growth}} - q^{\text{Fe:N}} \cdot \underbrace{V_{\text{dia}}^{N} \cdot PhyN_{\text{dia}}}_{\text{Diatom net growth}} \\
& - \underbrace{\kappa_{\text{Fe}} \cdot DetN \cdot Fe'}_{\text{Scavenging}}
\end{aligned}
\tag{5}
$$

$\epsilon_{\text{N}}^{\text{phy}}$ and $\epsilon_{\text{N}}^{\text{zoo}}$ are the excretion rates of organic nitrogen per day for phytoplankton and zooplankton. The parameter ρ_{DetN} is the temperature dependent degradation factor of detritus nitrogen concentration ($DetN$). $ZooN$ is the zooplankton nitrogen concentration and r_{zoo} is the zooplankton respiration rate per day. Net growth of plankton is described by the fourth term where V_{nano}^{N} and V_{dia}^{N} are the nitrogen assimilation rates for nanophytoplankton and diatoms, respectively. Finally, the last term depicts an iron

sink via scavenging with κ_{Fe} as the iron stability constant and Fe' as the concentration of free iron. Scavenging in this context is the net absorption into sinking organic matter.
Iron input is taken into account by a spatially and temporally varying iron flux for both the sediment and dust. This contrasts a proposition by Elrod et al. that suggests a constant supply of $2.65 \cdot 10^9$ mol DFe per year from the atmosphere and $2.67 \cdot 10^8$ mol DFe per year from the sediment [16].
Excretion from phytoplankton is depicted in the first term of Equation (5). For both nanophytoplankton and diatoms the summand contains a factor $f^{\mathrm{N:Cmax}}_{\mathrm{lim,\ \{nano,dia\}}}$ which limits the excretion of dissolved organic nitrogen and thus also the excretion of iron. The general limiting function for different metabolic process modulations is based on a slope s regulating the limitation rate and two parameters q_1 and q_2 which represent the limit value and current value, respectively:

$$f_{\mathrm{lim}}(s, q_1, q_2) = 1 - \exp\left(-s \cdot (\mid q_1 - q_2 \mid - (q_2 - q_1))^2\right) \tag{6}$$

Inserting the values for our special case leads us to a limiter function that is parameterized by $s^{\mathrm{N}}_{\mathrm{max}}$ and $q^{\mathrm{N:Cmax}}$ that can be found in Table 27. The third parameter $q^{\mathrm{N:C}}$ describes the current nitrogen to carbon ratio:

$$f^{\mathrm{N:Cmax}}_{\mathrm{lim,\ \{nano,dia\}}} \coloneqq f_{\mathrm{lim}}\left(s^{\mathrm{N}}_{\mathrm{max}}, q^{\mathrm{N:Cmax}}, q^{\mathrm{N:C}}\right) \tag{7}$$

Limiting functions are also used to end the release of nitrogen or limit an unregulated uptake of different nutrients in the process of assimilation. Figure 6 illustrates the intracellular regulation of nitrogen release and carbon update by the limiter function $f^{\mathrm{N:Cmax}}_{\mathrm{lim,\ nano}}$. Observe that a nitrogen to carbon quota of $16/106 \approx 0.151$ and higher causes a diminishing function output up to a complete limitation at a quota of 0.2. Such limitation is directly linked to the so-called Redfield ratio which describes the consistent atomic carbon to nitrogen ratio of 106 : 16 found on average in phytoplankton throughout ocean waters [42].

Zooplankton nitrogen concentration (ZooN):

$$\begin{aligned} SMS(ZooN) = & \underbrace{\gamma \cdot (G_{\mathrm{nano}} + G_{\mathrm{dia}})}_{\text{Grazing on phytoplankton}} \\ & - \underbrace{m_{\mathrm{zoo}} \cdot ZooN^2}_{\text{Zooplankton mortality}} \\ & - \underbrace{\epsilon^{\mathrm{zoo}}_{\mathrm{N}} \cdot ZooN}_{\text{DON excretion}} \end{aligned} \tag{8}$$

The second and third term describe nitrogen sinks with m_{zoo} as the zooplankton mortality rate, $ZooN$ as the nitrogen concentration in zooplankton and $\epsilon^{\mathrm{zoo}}_{\mathrm{N}}$ as the zooplankton excretion rate of organic nitrogen.
In the source term, we have a factor γ for the fraction of grazing flux to the zooplankton

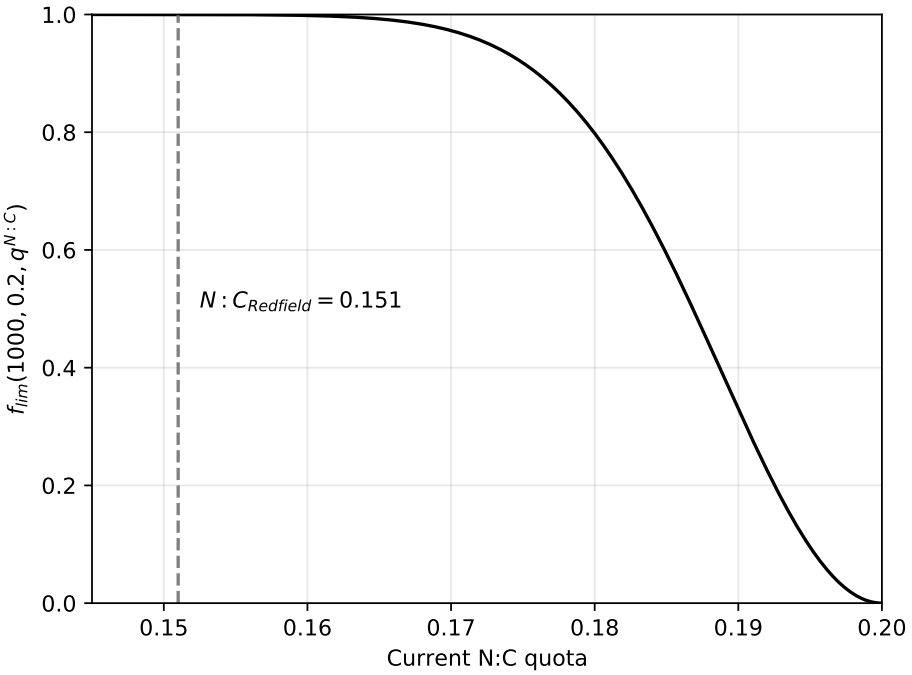


Figure 6: The limiter function $f^{\text{N:Cmax}}_{\text{lim, nano}}$ with parameter values $s^{\text{N}}_{\text{max}} = 1000$ and $q^{\text{N:Cmax}} = 0.2$.

pool as parameter in Table 24 multiplied by the grazing rates G_{nano} and G_{dia}. These grazing rates on either type of phytoplankton are calculated by their respective ratios of total grazing G_{tot}:

$$G_{\text{nano}} = G_{\text{tot}} \cdot \frac{PhyN_{\text{nano}}}{G'} \tag{9}$$

$$G_{\text{dia}} = G_{\text{tot}} \cdot \frac{PhyN_{\text{dia}} \cdot f^{\text{dia}}_{\text{Z}}}{G'} \tag{10}$$

$f^{\text{dia}}_{\text{Z}}$ is a constant of the relative grazing preference for diatoms and G' can be formulated as

$$G' = PhyN_{\text{nano}} + PhyN_{\text{dia}} \cdot f^{\text{dia}}_{\text{Z}} \quad . \tag{11}$$

The total grazing rate from Equation (9) and Equation (10) is calculated as follows:

$$G_{\text{tot}} = f_{\text{T}} \cdot G_{\text{max}} \cdot \frac{G'^2}{K_{\text{G}} + G'^2} \cdot ZooN \tag{12}$$

G_{max} is taken from Table 26 and represents the maximum grazing rate at 0°. It is on the one hand modulated by f_{T} (see Equation (4)) and by a ratio depending on a half-saturation constant for grazing loss on the other hand which can be found in Table 26.

Benthos:

Remember that the benthos is a seafloor compartment in *REcoM2* where nutrients are supplied by sinking detritus. As such it is also subject to the biological process of uptake and release of compounds. We exemplarily illustrate the formula for benthos silicon concentration (BenthosSi):

$$SMS(BenthosSi) = \underbrace{w_{\text{det}} \cdot DetSi}_{\text{Sinking detritus}} - \underbrace{\rho_{\text{Si}}^{\text{ben}} \cdot BenthosSi}_{\text{Remineralization}} \tag{13}$$

The sinking speed w_{det} is explained in Equation (1) and $\rho_{\text{Si}}^{\text{ben}}$ is a parameter found in Table 22 for the remineralization rate per day. $DetSi$ and $BenthosSi$ specify the concentration of silicon in detritus and benthos, respectively.
Note that the state variables in Table 19 regarding the nutrients in the benthos are given in $\frac{\text{mmol}}{m^2}$ as opposed to $\frac{\text{mmol}}{m^3}$ from ocean variables in Table 18. The nutrient concentrations in the benthos as boundary condition have been vertically integrated in order to act as a surface plane at the edge of three dimensional grid cells in the model. Remineralized nutrients are thus assumed to be released into the lowest grid cell. Even though only nitrogen, carbon, silicon and calcite are considered as state variable in the benthos, fluxes of alkalinity and iron from the benthos to the lowest layer of the water column are taken into account by variables from Table 21.

Alkalinity:

The benthos flux variable $BenF_{\text{Alk}}$ from Table 21 is calculated as follows:

$$BenF_{\text{Alk}} = \left(1 + \frac{1}{16}\right) \cdot \rho_{\text{N}}^{\text{ben}} \cdot BenthosN + 2 \cdot Diss_{\text{calc}} \cdot BenthosCalc \tag{14}$$

with $\rho_{\text{N}}^{\text{ben}}$ as remineralization rate of nitrogen in the benthos, $BenthosN$ and $BenthosCalc$ as nitrogen and calcite concentration in the benthos, respectively, and with $Diss_{\text{calc}}$ as the rate of calcium carbonate dissolution per day.
Alkalinity measures the ability of a volume of water to neutralise acids and is equal to the stoichiometric sum of the bases. With regard to our model, total alkalinity is affected by calcite, nitrogen and phosphate. All three compounds were considered in Equation (14); phosphate (P) as a nutrient is not included in *REcoM2* directly and is taken into account by the constant $P : N$ ratio of $1 : 16$. Hence, the factor of $\frac{1}{16}$ is added to the nitrogen remineralization term. The dissolution rate $Diss_{\text{calc}}$ of calcite on a length scale of $3500m$ is derived as follows [63]:

$$Diss_{\text{calc}} = \frac{w_{\text{z}}}{3500m} \tag{15}$$

Note that this derivation holds for the water column, but not the benthos. There, a constant dissolution rate is assumed, neglecting the dependence on the $CaCO_3$ saturation state. We define w_{z} as presented in Equation (1) for depth z. The SMS-function

for alkalinity (Alk) is formulated similarly to the benthos flux in Equation (14):

$$
\begin{aligned}
SMS(Alk) = &\left(1+\frac{1}{16}\right)\cdot\left(\underbrace{V^{\mathrm{N}}_{\mathrm{nano}}\cdot PhyC_{\mathrm{nano}}}_{\text{Nanoplankton N assimilation}} + \underbrace{V^{\mathrm{N}}_{\mathrm{dia}}\cdot PhyC_{\mathrm{dia}}}_{\text{Diatom N assimilation}}\right)\\
&-\left(1+\frac{1}{16}\right)\cdot\underbrace{\rho_{\mathrm{N}}\cdot f_{\mathrm{T}}\cdot DON}_{\text{DON Remineralization}}\\
&+2\cdot\underbrace{Diss_{\mathrm{calc}}\cdot DetCalc}_{\text{Detritus calcite dissolution}}\\
&-2\cdot\underbrace{\psi\cdot P_{\mathrm{nano}}\cdot PhyC_{\mathrm{nano}}}_{\text{Nanophytoplankton calcification}}
\end{aligned}
\tag{16}
$$

with $DetCalc$ as calcite concentration in detritus and ψ as nanophytoplankton production ratio of $CaCO_3$.
The term of $\frac{1}{16}$ again is added to compensate phosphate by a constant $1:16$ ratio. Equations for the nitrogen assimilation rates $V^{\mathrm{N}}_{\mathrm{nano}}$ and $V^{\mathrm{N}}_{\mathrm{dia}}$ for nanophytoplankton and diatoms, respectively, have been omitted. Upon dissolution of calcite from detritus, CO_3^{2-} is supplied to the water and increases alkalinity with two moles for each dissolved mole of calcium carbonate. Same applies to the reverse process of calcification which causes a multiplication by a factor of two in both summands.

3. Methodology

In this section we discuss the methodological approach for the Quantification of Uncertainties and Analysis of Sensitivities for Biogeochemical Ocean Simulations. Section 3.1 will present common methods used for Uncertainty Quantification. This includes the techniques of Monte Carlo simulations and Stochastic Collocation. In Section 3.2 the Bayesian framework will be introduced that engages the problem from a data driven perspective. After that we present the ideas of a variance-based Sensitivity Analysis in Section 3.3 where different measures of sensitivity and their computation techniques get displayed.

3.1. Uncertainty Quantification

This section focuses on the topic of Uncertainty Quantification. Note that this technique is usually tightly coupled with some form of Sensitivity Analysis which directly incorporates results from an Uncertainty Quantification. The main idea can be described as the quantification and successively the propagation of uncertainties through the model in order to determine its sensitivities with respect to these uncertainties.
Sources of uncertainties can be multifold, especially for complex models. In Section 1.1.2 we presented a distinction between uncertainties arising from either the model setup, numerical limitations of computers or model input. Since the scope of this thesis does not cover a deep analysis or even questioning of the underlying biogeochemical model and accuracy of computers, we project all uncertainties onto the model inputs. The *REcoM2* model has been put into action before and provided reasonable results with a predefined set of input parameters that simulate conditions in the Bermuda region. We hence assume that all parameters are already tweaked to feasible values to a qualified extent. Uncertainties in the inputs are therefore limited to a relatively small interval. An additional limiting factor are physical bounds of parameters. Consider for example the light harvesting efficiency α for both types of phytoplankton (see Table 25), which naturally has to be non-negative.
In the following we examine the Monte Carlo method, where distribution functions dictate the range and probability of values for uncertain parameters. However, independent sampling of a uniformly random value for each parameter neglects possible dependencies or correlations between two or more parameters. Even though the detritus sinking speed w_0 (see Table 24), for instance, will most likely be uncorrelated to a parameter like the chlorophyll degradation rate deg_{Chl} (see Table 22), we cannot easily make definite statements about each subset of parameters. A clear dependency can at least be derived for parameters regarding the limitation functions in Table 27, where for example $s^{\text{Si}}_{\text{min}} \leq s^{\text{Si}}_{\text{max}}$ must hold. In order to address this problem of parameter dependency, we will further explore a method in the Bayesian framework. Note that the application of this method is beyond the scope of this thesis and only serves as an introduction to potential future work. Before that we offer an alternative to classic Monte Carlo simulations which is usually referred to as Stochastic Collocation and which employs a deterministic sampling technique.

3.1.1. Monte Carlo

Deterministic numerical integration encounters problems if the underlying function has many variables. The number of function evaluations required increases exponentially with the number of dimensions, which is sometimes called the "curse of dimensionality". For instance, assuming an adequate accuracy for one dimension is achieved by 100 evaluations, then all 51 degrees of freedom in *REcoM2* (see Tables 22-27) would demand 100^{51} evaluations, which is not feasible at all for such a complex model. The Monte Carlo method breaks out of these dimensional constraint by solving the multi-dimensional definite integral for a quantity of interest probabilistically. Observe that such modification will turn the inherently deterministic model into a stochastic one.
Let $f(\vec{X})$ be the numerical model function with the input parameters $\vec{X}$. A function for a quantity of interest is generically designed by the user. An integral often yields a desired value and such function can be mathematically formulated as

$$\int_{\Omega_m} f(\vec{X})\, d\vec{X} \tag{17}$$

where $\Omega_m \subseteq \mathbb{R}^m$ is the parameter domain in the m-dimensional hypercube with $m = |\vec{X}|$.
Remember that we roam in the area of Uncertainty Quantification where some values of $\vec{X}$ are affected by uncertainty. We thus split the input into a new vector $\vec{\chi}$ of undisturbed parameters and a vector $\vec{\xi}$ of parameters susceptible to uncertainties, i.e. $\vec{X} = (\vec{\chi}, \vec{\xi})$. In order to describe the domain of uncertainty, we define a probability space for the parameters of $\vec{\xi}$ by assigning suitable probability density functions $r(\vec{\xi})$. The question of a most adequate distribution for $\vec{\xi}$, i.e. function r, cannot easily be answered, but the focus lies on continuous probability distributions because the parameters of *REcoM2* are represented by floating-point values. Since each parameter of *REcoM2* has some predefined default setting t and uncertainties are supposed to address relatively small deviations, it is plausible to employ a probability density function where the statistical properties of mean, median or mode lie close to t and where values are more improbable the more they diverge from t. Such intuition naturally excludes uniform or exponential distributions for example. A commonly used distribution for this purpose is the normal distribution. With a normal distribution $\mathcal{N}(1, \sigma^2)$ for some standard deviation σ, we would center its bell curve above the normalizing factor of one such that uncertainties are realized by a factor multiplication of t with a normally distributed variable. However, the question of how to handle negative values remains because negative parameters are explicitly not allowed and could potentially even crash the execution of a model run. For this reason, we will use the lognormal distribution $\mathcal{LN}(0, 0.125)$ throughout this project. A random variable A is lognormally distributed if and only if $A = e^B$ for a normally distributed random variable B. Lognormally distributed random variables thus cannot be negative. The downside is a skewness of the probability density function (pdf) to the right. Notice that our predefined variance of $\sigma^2 = 0.125$ above results in a pdf which closely resembles that of a normal distribution (see Figure 7). Smaller values for σ^2 would in theory conform to statistics of a normal

distribution even more, but we stick to our setting for the sake of a broader uncertainty margin.

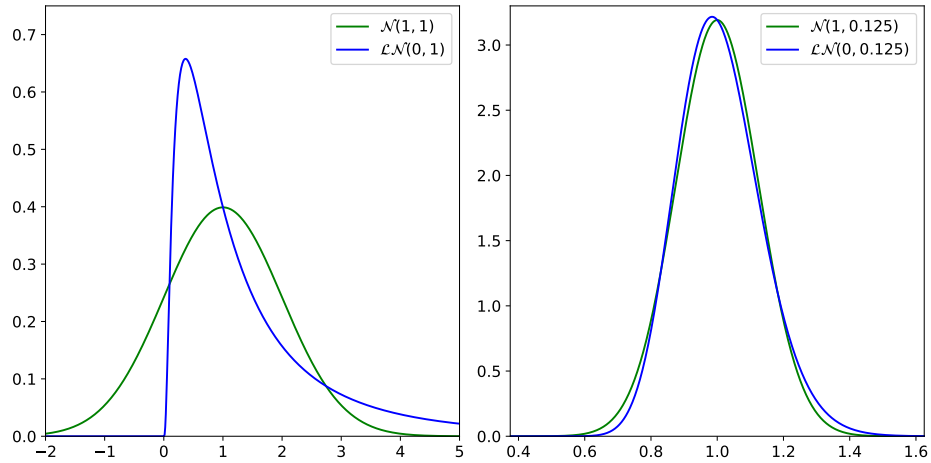


Figure 7: Comparison of probability density functions (pdf) from a normal distribution and a lognormal distribution with variances of 1 (left) and 0.125 (right), respectively. Note the resemblence of the curves on the right and that the mean of $\mathcal{LN}(0, 0.125)$ lies close enough to one for our purpose with $\exp\left(\frac{\sigma^2}{2}\right) = \exp\left(\frac{0.125}{2}\right) \approx 1.064$.

Respective probability functions $r(\vec{\xi})$ enable us to reformulate the expected value of $f(\vec{X})$ with regard to $\vec{X}$ in order to fit our needs. The calculation for a quantity of interest, such as the expectation of chlorophyll concentration in output data of *REcoM2* , can be written as

$$\mathbb{E}\left[f(\vec{\chi}, \vec{\xi})\right] = \int_{\Omega_d} f(\vec{\chi}, \vec{\xi}) r(\vec{\xi})\, d\vec{\xi} =: I \quad . \tag{18}$$

Observe that in contrast to Equation (17), we integrate over the vector $\vec{\xi} = \{\xi_1, \ldots, \xi_d\}$. The integral bounds are therefore only d-dimensional here. However, without loss of generality we assume in the following for convenience that $\Omega_d \subseteq [0, 1]^d$ instead of $\mathbb{R}^d$ since all parameters can in theory be normalized to the d-dimensional unit hypercube. The Monte Carlo approach now samples points uniformly at random on unit hypercube Ω_d. Since computers are inherently deterministic machines, generating random numbers is a difficult task. In this thesis we employ the pseudo-random number generator that is embedded in the core Python library. Its documentation states the following [19]:

> "Almost all module functions depend on the basic function *random()*, which generates a random float uniformly in the semi-open range $[0.0, 1.0)$. Python uses the Mersenne Twister as the core generator. It

produces 53-bit precision floats and has a period of $2^{19937} - 1$. The underlying implementation in C is both fast and threadsafe. The Mersenne Twister is one of the most extensively tested random number generators in existence."

Given a number of N generated samples $\vec{\xi^1}, \ldots, \vec{\xi^N}$, Equation (18) can be approximated by

$$I \approx \mathcal{C}_N \equiv \frac{1}{N} \sum_{n=1}^{N} f(\vec{\chi}, \vec{\xi^n}) \tag{19}$$

where $f(\vec{\chi}, \vec{\xi^n})$ is the model function for the respective sample. Recall that $\vec{\chi}$ is fixed. As long as the sequence $\vec{\xi^1}, \ldots, \vec{\xi^N}$ only consists of independent and identically distributed random parameters, the law of large numbers applies:

$$\lim_{N \to \infty} \mathcal{C}_N = I \tag{20}$$

In order to assess the performance of the Monte Carlo approach, we need to explore the convergence rate of $\mathcal{C}_N$ since an equivalence as shown in Equation (20) cannot be reached in practice. The error of $\mathcal{C}_N$ can be estimated by calculating the sample variance σ_N^2 in Equation (21) and using it to derive the variance of $\mathcal{C}_N$ in Equation (22) as follows:

$$\mathrm{Var}(f) \approx \sigma_N^2 = \frac{1}{N-1} \sum_{i=1}^{N} \left(f(\vec{\chi}, \vec{\xi^i}) r(\vec{\xi^i}) - \frac{1}{N} \sum_{j=1}^{N} f(\vec{\chi}, \vec{\xi^j}) \right)^2 \tag{21}$$

This leads to

$$\mathrm{Var}(\mathcal{C}_N) = \frac{1}{N^2} \sum_{i=1}^{N} \sigma_N^2 = \frac{\sigma_N^2}{N} \quad . \tag{22}$$

The bounded sequence $\{\sigma_1^2, \ldots, \sigma_N^2\}$ ensures a variance that decreases asymptotically to zero. Finally, we can derive the standard error of the mean for $\mathcal{C}_N$ with

$$\varepsilon(\mathcal{C}_N) \approx \sqrt{\mathrm{Var}(\mathcal{C}_N)} = \frac{\sigma_N}{\sqrt{N}} \tag{23}$$

which decreases as $\frac{1}{\sqrt{N}}$. Monte Carlo sampling thus converges with a rate of $\mathcal{O}(N^{-1/2})$ where $\mathcal{O}(\cdot)$ denotes the upper bound for the growth rate of a function. Observe that this method does indeed not depend on the dimension of the probability space Ω_d. However, enhancing the accuracy by one digit increases the number of required simulations by a factor of 100.

3.1.2. Quasi Monte Carlo

In the last section we introduced the standard Monte Carlo method and proved its convergence rate of $\mathcal{O}(N^{-1/2})$. Morokoff and Caflisch presented a way that maintains the

main sampling idea, but replaced pseudo-random numbers with quasi-random numbers generated by so-called low-frequency sequences [33]. They compared three such sequences, the Sobol, the Halton and the Faure sequence. For numerical problems with more than six parameters, the Sobol sequence performed best and hence deserves our attention. Generally, these sequences serve as reference points for the creation of a hypercube filled with a set of points such that void regions are avoided. Figure 8 illustrates the difference between numbers generated by pseudo-random generators and an algorithm which employs the Sobol sequence. More evenly distributed points in the

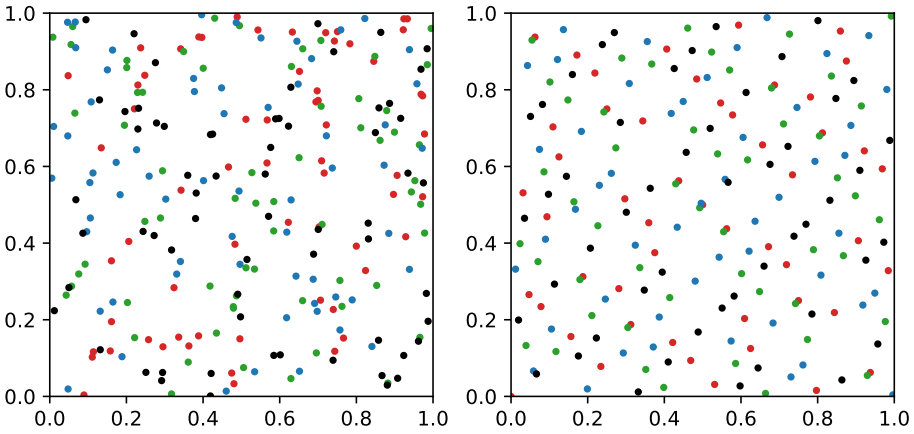


Figure 8: Unit planes with 256 consecutive points generated by a pseudo-random generator (left) and a generator based on a Sobol sequence (right). The colors indicate the indices within the sequence (red: 1-64, green: 65-128, blue: 129-192, black: 193-256).

hypercube enable a potentially faster convergence rate than that of classic Monte Carlo simulations. Caflisch et al. showed that using the Sobol sequence for the generation of quasi-random numbers resulted in a convergence rate of $\mathcal{O}((\log N)^d N^{-1})$ where d is the dimension of the considered probability space [9]. This means that for a small number of dimensions, the quasi-Monte Carlo approach needs only a fraction of simulations for an additional digit of accuracy as compared to pure Monte Carlo. The drawback becomes apparent if d is larger and the number of samples N remains relatively low, where the classic Monte Carlo method outperforms the quasi-Monte Carlo method in terms of convergence.

3.1.3. Stochastic Collocation

Unlike previous methods, the Stochastic Collocation method does not reside in the realm of Monte Carlo simulations. However, it still can be described as a sampling-based method. The major difference is the fact that Stochastic Collocation employs a deterministic sampling approach and not a probabilistic one.
Instead of generating samples based on probability distributions and using them to

calculate an ensemble average like in Equation (19), Stochstic Collocation deterministically predefines collocation points and then computes the solution of the underlying function at these points. By strategically assigning weights as coefficiants to the nodes, we are able to approximate the expectation value from Equation (18) by

$$\mathbb{E}\left[f(\vec{\chi},\vec{\xi})\right] = \int_{\Omega_d} f(\vec{\chi},\vec{\xi}) r(\vec{\xi})\, d\vec{\xi} \approx \sum_{i=1}^{n} \omega_i f(u_i) \tag{24}$$

where ω_i are the weights and $f(u_i)$ are the function evaluations at collocation point u_i with $1 \leq i \leq n$ [40]. Note that each collocation point u_i is w.l.o.g. defined on the unit hypercube Ω_d as well. Same approach can be employed for the calculation of the function variance by

$$\mathbb{V}\left[f(\vec{\chi},\vec{\xi})\right] = \mathbb{E}\left[f(\vec{\chi},\vec{\xi})^2\right] - \mathbb{E}\left[f(\vec{\chi},\vec{\xi})\right]^2 \approx \sum_{i=1}^{n} \omega_i f(u_i)^2 - \left(\sum_{i=1}^{n} \omega_i f(u_i)\right)^2 \quad . \tag{25}$$

A set of collocation points usually consists of the nodes of a quadrature rule in multidimensional space defined on the unit hypercube such that one can use the integration rule defined by the quadrature to calculate statistics for a quantity of interest. Multidimensional quadratures rules are sometimes called cubatures rules and are characterized by its degree. A rule of degree d, for example, indicates that an integral is exact as long as the integrand is any multivariate polynomial of degree less or equal to d.
The selection of a suitable cubature rule is essential for Stochastic Collocation because the underlying function has to be calculated for every cubature point and the number of these points increases exponentially with the dimension d of the probability space Ω_d. Gauss quadratures usually provide the optimal choice for $d = 1$, but the challenge is in the multidimensional space with $d > 1$.
One choice is to apply the tensor product to one-dimensional nodes, for example the aforementioned Gauss quadratures. Mathelin and Hussaini used this approach among others in order to conveniently generalize properties of one-dimensional integration [31]. However, the number of points still grow rapidly in multidimensional probability spaces, and they concluded that the tensor product approach is not feasible for dimensions $d > 5$ in most applications [4].
Another way to construct efficient cubature rules are sparse grids which are based on the Smolyak algorithm [51]. They are a subset of the full tensor product grids and the points are chosen strategically in a way that the advantageous approximation properties for the one-dimensional probability space are preserved and for the multidimensional case as much as possible [62]. Figure 9 shows an example for tensor product grids and sparse grids. Latter clearly contain much less number of points such that they can provide the framework to conduct a Stochastic Collocation approximation in a high dimensional setting. Observe that the sparse grid still contains the same set of points along the dimensional axes. According to Smith the quadrature error is of order

$$\mathcal{O}(N^{-\alpha}(\log N)^{(d-1)(\alpha+1)}) \ , \tag{26}$$

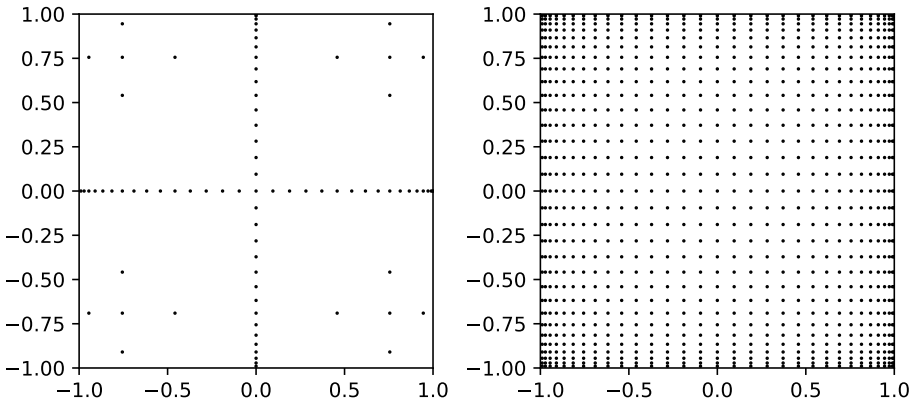


Figure 9: Examples of collocation points in two-dimensional space. Left: Sparse grid. Right: Tensor product grid.

with N the number of collocation points and α a constant that is dependent on the smoothness of the function [50]. The node count (number of collocation points) is dependent on the node type. In this work we employ the Clenshaw Curtis (CC) nodes. These nodes are typically defined on the interval $[-1, 1]^d$ and describe the extrema of the Chebyshev polynomials [14]. For a desired level l of refinement the nodes can be calculated by

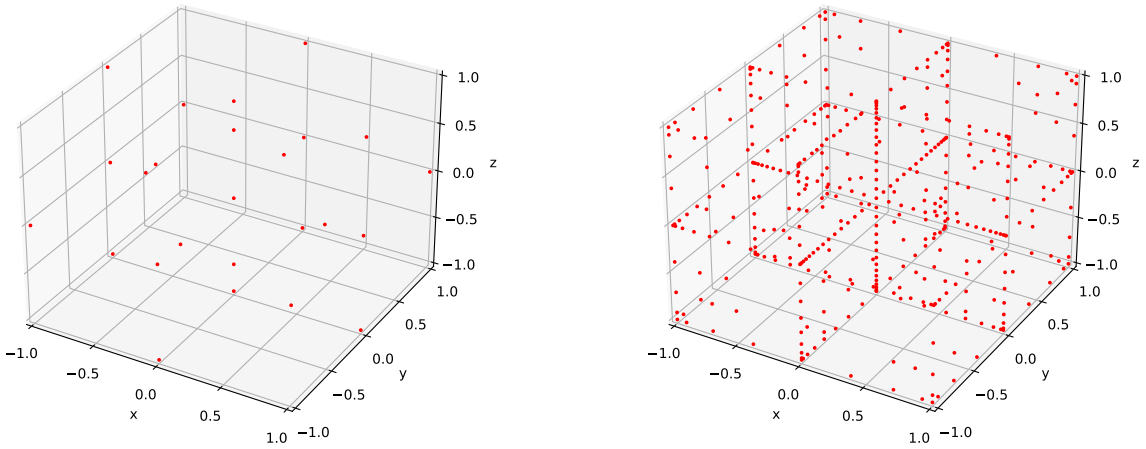
$$Q_l^r = -\cos\frac{\pi(r-1)}{R_l - 1} \quad , 1 \leq r \leq R_l \tag{27}$$

where $R_1 = 1$ and $R_l = 2^{l-1} + 1$ for $l > 1$. Figure 10 illustrates the arrangement of CC nodes with different refinement levels in three dimensional space. It is interesting to observe the effect of increasing refinement levels (Figure 10a and 10b) and increasing dimension (Figure 9(left) and 10b) on node count and nestedness.
As the Clenshaw Curtis node type yields nodes on the interval $[-1, 1]$ and thus would not be a suitable quadrature rule for unbounded probability density functions like for example the normal distribution, we use an inverse cumulative distribution function transform to translate the set of nodes to $(-\infty, \infty)$ [58].

3.2. Bayesian Method

In the last sections we discussed Uncertainty Quantification methods which are based on forward uncertainty propagation. In these cases the uncertainties are propagated through the model to predict the overall uncertainty for a quantity of interest (see Figure 11).

Equation (18) already formulated this system mathematically. For the sake of simplicity, we now omit a distinction between certain and uncertain parameters and rather use $\vec{\xi}$ as (possibly) uncertain input parameters in general. Moreover, an explicit functional is also left out in the following and is replaced by the notion of $g(\cdot)$. We reach



(a) $d = 3, l = 2$ (b) $d = 3, l = 5$

Figure 10: Visualization of sparse grids with nested Clenshaw Curtis nodes.

the rephrased formula

$$I(\vec{\xi}) \equiv I \coloneqq g(f(\vec{\xi})) \tag{28}$$

where $f(\cdot)$ still describes the underlying model function.
Our goal now is to consider an inverse problem, that is, examining observational data y and finding input parameters $\vec{\xi}^*$ such that the equation

$$y = g(f(\vec{\xi}^*)) + \epsilon \tag{29}$$

is satisfied with the observation error ϵ as small as possible. Figure 12 illustrates the paradigm shift where the input parameter $\vec{\xi}^*$ is under investigation.
Solving for equality is unrealistic in practice due to its ill-posedness according to Hadamard [20]. Techniques like the Tikhonov Regularization hence aim for the minimization of the residual for the Euclidian norm $||\cdot||_2$ [56]. Unfortunately, this method is not useful for our purposes because we want to extract additional information that helps us quantifying uncertainties.
Remember that we acknowledged the difficulty of finding suitable probability density functions for each parameter and detecting mutual dependencies of parameters with regard to Monte Carlo methods. Using a Bayesian approach will help us overcome this obstacle by fine-tuning the probability distribution of input parameters based on observational data.
Since randomness perturbs Equation (29), we now consider $\vec{\xi}$ to be a random variable. The Bayesian framework facilitates the acquisition of knowledge about the entire distribution of parameters $\vec{\xi}$ given that we have observed y. In statistical terms we can denote this as the conditional $\mathbb{P}(\vec{\xi} \mid y)$ which poses as the posterior distribution. It can be attained with help of the fundamental Bayesian Theorem:

$$\mathbb{P}(\vec{\xi} \mid y) = \frac{\mathbb{P}(y \mid \vec{\xi})\mathbb{P}(\vec{\xi})}{\mathbb{P}(y)} \tag{30}$$

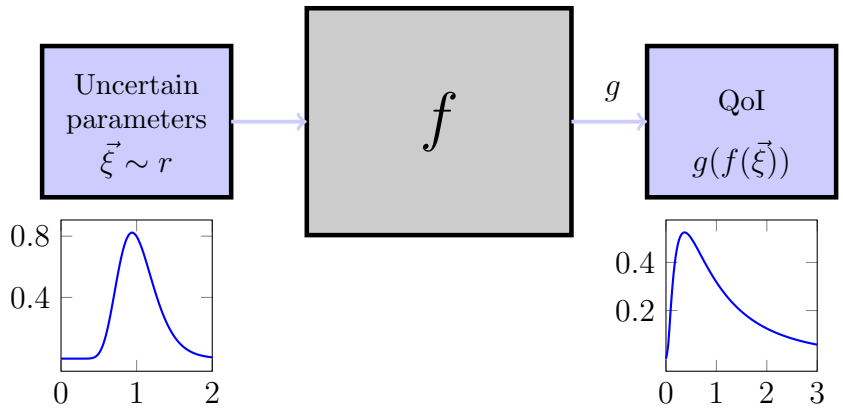


Figure 11: Setup for a forward uncertainty propagation model like the (quasi)-Monte Carlo Simulation.

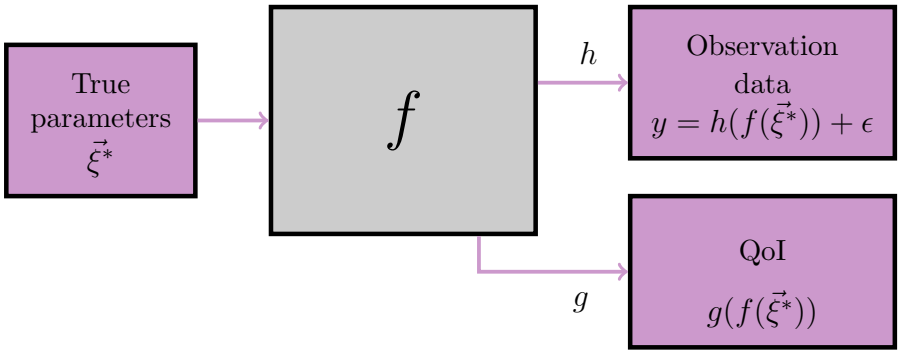


Figure 12: UQ model setup assuming that the actual parameters $\vec{\xi}^*$ are known. Note that the obseration data y is affected by some error ϵ.

The probability of obeserved data is independent of $\vec{\xi}$ and not relevant for our examination which leaves us with

$$\mathbb{P}(\vec{\xi} \mid y) \sim \mathbb{P}(y \mid \vec{\xi})\mathbb{P}(\vec{\xi}) \quad . \tag{31}$$

In this context $\mathbb{P}(\vec{\xi})$ is the a priori distribution and describes our initial descriptions of probability density function for the input parameters. The likelihood $\mathbb{P}(y \mid \vec{\xi})$ is the probability of observed data given that the input vector is fixed. Assuming that we employ a multivariate normal distribution for an Bayesian approach and define for the measurement error that $\epsilon \sim \mathcal{N}(0, \Sigma)$, we can concretely deduce it with Equation (29):

$$\begin{aligned} & g(f(\vec{\xi}^*)) - y = \epsilon \sim \mathcal{N}(0, \Sigma) \\ \Leftrightarrow\ & y \sim \mathcal{N}\left(g(f(\vec{\xi}^*)), \Sigma\right) \end{aligned} \tag{32}$$

Once we have incorporated our findings regarding the likelihood from Equation (32) into the posterior distribution, we can finally close the feedback loop for the readjustment of the uncertainty distributions of $\vec{\xi}$. Figure 13 illustrates this cycle.

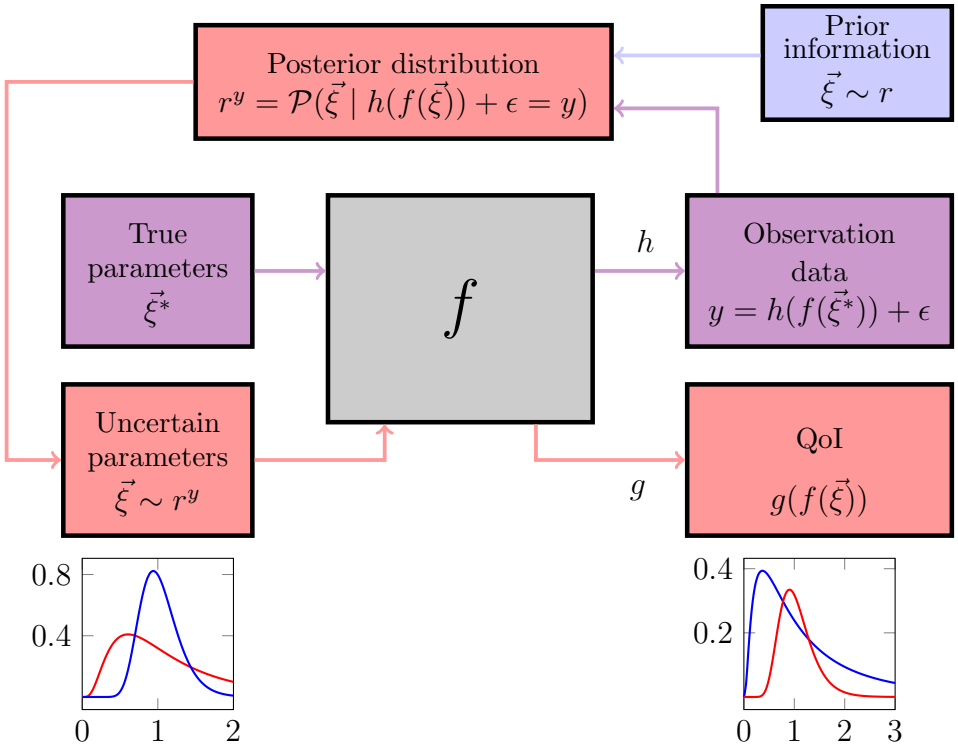


Figure 13: Inverse UQ model with the Bayesian method. The feedback loop allows refining prior distributions by a type of data assimilation.

We have to stress that this general Bayesian framework itself is not a standalone method to generate samples, but rather an approach to refine a priori defined probability density functions by a posteriori distributions r^y. With a sampling based method like Monte Carlo, for example, samples can be generated from r^y such that $\vec{\xi} \sim r^y$.
Note that we do not employ the Bayesian framework for this thesis. The introduction above, however, aims at giving the reader an insight of potentially useful work that may be tackled in the future.

3.3. Sensitivity Analysis

We are now shifting our attention from a pure examination of uncertainties for input parameters to a more comprehensive view of model statistics. Nevertheless, Uncertainty Quantification is a key prerequisite for a Sensitivity Analysis where the impacts of uncertainties with regard to the model output are investigated.

REcoM2 as a complex and non-linear model is preferably analysed with global Sensitivity Analysis methods [8]. In this thesis we will employ a variance-based Sensitivity Analysis which is a member of those global approaches.

3.3.1. Variance-Based SA

The variance-based Sensitivity Analysis is based on two mathematical observations:

1. **Law of Total Variance** [61]: Remember that our methods of Uncertainty Quantification have turned the model into a stochastic one because both the system response y and input parameters $\vec{\xi} = (\xi_1, \ldots, \xi_n)$ are treated as random variables. The law now states that

$$\mathbb{V}[y] = \mathbb{V}\left[\mathbb{E}[y \mid \xi_i]\right] + \mathbb{E}[\mathbb{V}[y \mid \xi_i]] \tag{33}$$

   where $\mathbb{V}[\cdot]$ ($\mathbb{V}[\cdot \mid \cdot]$) and $\mathbb{E}[\cdot]$ ($\mathbb{E}[\cdot \mid \cdot]$) denote the (conditional) variance and (conditional) expected value, respectively. In other words, the total variance $\mathbb{V}[y]$ can be decomposed into the sum of two terms: One measures the variance between the conditional means given the inputs ξ_i. The second term describes the mean of the conditional variances which is sometimes referred to as the residual.

2. **High-Dimensional Model Representation** [52]: Sobol showed that any function of the type $y = f(\xi_1, \xi_2, \ldots, \xi_n)$ can be decomposed into the following terms where $1 \leq i_1 < \cdots < i_s \leq n$:

$$y = f(\xi_1, \xi_2, \ldots, \xi_n) = \mathbb{E}[y] + \sum_{s=1}^{n} \sum_{i_1 < \cdots < i_s}^{n} z_{i_1 \ldots i_s}(\xi_{i_1}, \ldots, \xi_{i_s}) \quad . \tag{34}$$

   Note that Equation (34) contains 2^n summands that can be disentangled such that

$$y = \mathbb{E}[y] + \sum_{i=1}^{n} z_i(\xi_i) + \sum_{i<j} z_{ij}(\xi_i, \xi_j) + \cdots + z_{12\ldots n}(\xi_1, \xi_2, \ldots, \xi_n) \tag{35}$$

   where

$$z_i(\xi_i) = \mathbb{E}[y \mid \xi_i] - \mathbb{E}[y] \quad . \tag{36}$$

$$z_{ij}(\xi_i, \xi_j) = \mathbb{E}[y \mid \xi_i, \xi_j] - \mathbb{E}[y \mid \xi_i] - \mathbb{E}[y \mid \xi_j] - \mathbb{E}[y] \quad . \tag{37}$$

   All terms of higher order up to $z_{12\ldots n}(\xi_1, \xi_2, \ldots, \xi_n)$ are derived analogously. The $z_i(\xi_i)$ terms are called the main effects (or first-order interactions), the $z_{ij}(\xi_i, \xi_j)$ terms are denoted as second-order interactions and so on.
   Note that these terms are dependent on the probability distribution of the uncertain parameters which is illustrated by Example 3.1.

Example 3.1. Let the considered model be $f(\xi_1, \xi_2) = \xi_1$ and $y = f(\xi_1, \xi_2)$. We have $\mathbb{E}[y] = \mathbb{E}[\xi_1]$ and thus

$$\begin{aligned} z_1(\xi_1) &= \mathbb{E}[y \mid \xi_1] - \mathbb{E}[y] \\ &= \mathbb{E}[f(\xi_1, \xi_2) \mid \xi_1] - \mathbb{E}[f(\xi_1, \xi_2)] \\ &= \xi_1 - \mathbb{E}[\xi_1] \end{aligned} \tag{38}$$

If the distributions of ξ_1 and ξ_2 are independent then $z_2(\xi_2) = 0$ and $z_{12}(\xi_1, \xi_2) = 0$. In this case the representation reflects the structure of the model function itself with a linear effect of ξ_1 without interactions with ξ_2. If however both parameters are not independent, we have the general case of

$$z_2(\xi_2) = \mathbb{E}[\xi_1 \mid \xi_2] - \mathbb{E}[\xi_1] = -z_{12}(\xi_1, \xi_2)$$

which is not necessarily zero.

3.3.2. Sensitivity Measures

In the previous section we prepared the mathematical framework to extract different kind of sensitivity measures. It is up to the user and dependent on the application which measures fit best. We present some popular sensitivity measures that were first presented by Saltelli et al. [46] and later extended by Marzban [30]. See Table 1 for a list of selected sensitivity measures.
The first measure is given by $\mathbb{E}[\mathbb{V}[y] - \mathbb{V}[y \mid \xi_i]]$ and is motivated by an expected reduction in the variance of the output. It is equal to $\mathbb{V}[y] - \mathbb{E}[\mathbb{V}[y \mid \xi_i]]$ and according to Equation (33) it can be rewritten as

$$V_i \coloneqq \mathbb{V}[\mathbb{E}[y \mid \xi_i]] \quad . \tag{39}$$

Similar measures for an expected reduction of output uncertainty with multiple fixed parameters can be obtained analogously. For example, it follows from Equations (34)-(37) that a measure complementary to V_i can be deduced given the premise that we knew the true value of two inputs:

$$V_{i+j} \coloneqq \mathbb{V}[\mathbb{E}[y \mid \xi_i, \xi_j]] = V_i + V_j + \mathbb{V}[z_{ij}(\xi_i, \xi_j)] \tag{40}$$

For the case that $z_i(\xi_i)$ and $z_j(\xi_j)$ are mutually independent we can assess the level of interaction between ξ_i and ξ_j with

$$V_{ij} \coloneqq \mathbb{V}[z_{ij}(\xi_i, \xi_j)] \quad . \tag{41}$$

In contrast to previous measures, observing the uncertainty remaining in the output while given all input parameters, except ξ_i, can be expressed by

$$V_{Ti} \coloneqq \mathbb{V}[y] - \mathbb{V}\left[\mathbb{E}\left[y \mid \vec{\xi}_{\sim i}\right]\right] \tag{42}$$

where $\vec{\xi}_{\sim i}$ denotes the vector of all input parameters $\vec{\xi}$ except ξ_i which results in the vector $\vec{\xi}_{\sim i} = (\xi_1, \ldots, \xi_{i-1}, \xi_{i+1}, \ldots, \xi_n)$.
Usually, the measures V_i and V_{Ti} are converted to scale invariant proportions, respectively:

$$S_i \coloneqq \frac{V_i}{\mathbb{V}[y]} \tag{43}$$

$$S_{Ti} \coloneqq \frac{V_{Ti}}{\mathbb{V}[y]} \tag{44}$$

Sensitivity Measure	Description
$V_i = \mathbb{V}[\mathbb{E}[y \mid \xi_i]]$	Reduction in uncertainty of y, given ξ_i
$V_{ij} = \mathbb{V}[z_{ij}(\xi_i, \xi_j)]$	Interaction between ξ_i and ξ_j (if $z_i(\xi_i)$ and $z_j(\xi_j)$ are mutually independent)
$V_{Ti} = \mathbb{V}[y] - \mathbb{V}\left[\mathbb{E}\left[y \mid \vec{\xi}_{\sim i}\right]\right]$	Uncertainty remaining in y, given every parameter except ξ_i
$S_i = V_i / \mathbb{V}[y]$	Normalized first order index
$S_{Ti} = V_{Ti} / \mathbb{V}[y]$	Normalized total effect index

Table 1: Sensitivity measures and their effect on output uncertainty.

3.3.3. Sensitivity Calculation

Saltelli presents different approaches for the calculation of both S_i and S_{Ti}. All techniques, however, share the same core structure of two independently created sample matrices $\mathbf{A}$ and $\mathbf{B}$ that need to be evaluated by the underlying model function. These matrices consist of N rows and d columns such that each row represents one generated sample in the d-dimensional probability space of uncertain parameters. From $\mathbf{A}$ and $\mathbf{B}$ we introduce the matrix ${\mathbf{A}_\mathbf{B}}^i$ (${\mathbf{B}_\mathbf{A}}^i$) where all columns are from $\mathbf{A}$ ($\mathbf{B}$) except the i-th column which is from $\mathbf{B}$ ($\mathbf{A}$). Figure 14 illustrates the generation of those matrices. Sensitivity index S_i can be calculated by dividing

$$V_i \approx \frac{1}{N} \sum_{j=1}^{N} f(\mathbf{B})_j (f({\mathbf{A}_\mathbf{B}}^i)_j - f(\mathbf{A})_j) \tag{45}$$

by the normalizing model variance $\mathbb{V}[y]$ where $f(\mathbb{R}^{N \times d})_j$ denotes the model evaluation with the input values of the j-th row of the sample matrix $\mathbb{R}^{N \times d}$. Observe that only the triplet $\mathbf{A}$, $\mathbf{B}$ and ${\mathbf{A}_\mathbf{B}}^i$ is required for this estimate by Saltelli. Jansen proposed an even better estimator that does not require the evaluations from sample matrix $\mathbf{A}$, but the estimated model variance $\mathbb{V}[y]$ [24]:

$$V_i \approx \mathbb{V}[y] - \frac{1}{2N} \sum_{j=1}^{N} (f(\mathbf{B})_j - f({\mathbf{A}_\mathbf{B}}^i)_j) \tag{46}$$

$\mathbf{A}$			$\mathbf{B}$			${\mathbf{A}_\mathbf{B}}^1$		
3.14	0.54	7.40	3.54	0.77	7.81	3.54	0.54	7.40
2.95	0.41	6.57	3.11	1.01	7.47	3.11	0.41	6.57
3.04	0.98	7.32	2.78	0.04	7.01	2.78	0.98	7.32
3.15	0.56	7.89	3.71	0.28	8.14	3.71	0.56	7.89
3.33	0.67	7.03	3.34	0.71	7.55	3.34	0.67	7.03

Figure 14: Exemplary sample matrices $\mathbf{A}$, $\mathbf{B}$ and ${\mathbf{A}_\mathbf{B}}^1$ with $N = 5$ and $d = 3$. Matrices ${\mathbf{A}_\mathbf{B}}^2$ and ${\mathbf{A}_\mathbf{B}}^3$ are generated analogously to ${\mathbf{A}_\mathbf{B}}^1$.

Sensitivity S_{Ti} can be obtained by dividing the following estimation of V_{Ti} by $\mathbb{V}[y]$:

$$V_{Ti} \approx \frac{1}{2N} \sum_{j=1}^{N} (f(\mathbf{A})_j - f({\mathbf{A}_\mathbf{B}}^i)_j)^2 \tag{47}$$

We will use the estimators from Equations 46 and 47 in our work which allows us to avoid additional model simulations of ${\mathbf{B}_\mathbf{A}}^i$. Model variance $\mathbb{V}[y]$ is generally estimated by appending $\mathbf{A}$ and $\mathbf{B}$ such that the variance is dependent on both sample matrices.

It is important to point out that previous formulas for the calculation of sensitivities are only estimates and are not exact with regard to Equations (39) and (42). Even more so, the estimators we use may for example yield negative values which is naturally not possible. Calculating the sensitivities directly with exact formulas would, however, require a simulation overhead of order $\mathcal{O}(N^2)$ with the sample size N. Degrees of accuracy are thus commonly traded in with a linear growth rate of simulations instead. In order to still be able to assess the quality and robustness of our results, we go beyond an error estimate by Sobol that merely relies on the underlying sampling variability [53]. The *bootstrap subsampling* technique formulated by Archer et al. [2] allows us to evaluate the sensitivities based on already obtained data. Its idea is to choose a number of subsamples from the original data set and to calculate the sensitivities on these subsamples. If the estimate is already sufficiently accurate the sensitivities computed by the subsamples will hopefully not diverge greatly from this value. The span of divergence is then embedded in a confidence interval which provides an educated guess for the precision of the sensitivities.

An open question remains the adequate number of times the subsampling should be executed (number of replicas). For our work we use 10000 replicas which has proven effective in literature [2] [46]. As for the size of the subsample set, we found that $N/4$ subsampling points yield robust results.

4. Implementation

In this section we present an implementation of methodological approaches discussed in Section 3. The overall layout of the execution pipeline will be illustrated in Section 4.1. This pipeline consists of roughly three steps that are modularized in mutually independent algorithms. Even though these steps are independent of one another in terms of their execution, they still have to be run consecutively because the input of a step is dependent on the output of the previous one.
Section 4.2 and 4.3 address our implementations for the Uncertainty Quantification methods as the first step. *REcoM2* as the underlying model is an external implementation that is also executed on remote computer systems. Section 4.4 will therefore focus on the interface that we have established with that system. In order to test our implementation and to avoid a time-delayed and computational expensive simulation, we furthermore demonstrate a synthetic test model in Section 4.5. As the final step, Section 4.6 presents our implementation of a variance-based Sensitivity Analysis. Before starting a case study we also present an empirical validation of the implemented methods in Section 4.7.

4.1. Architecture

Goal of this thesis is the contribution of profound results to the investigation of highly sensitive parameters in *REcoM2* and to answer the dual question of which parameters can be neglected. Our variance-based Sensitivity Analysis requires a preceding Uncertainty Quantification and its trailing model execution with multiple parameter configurations for that task. This rough setup is illustrated by Figure 15. As stated above, these three steps have to be run consecutively. Note that the settings file `config.json` serves as the overall configuration file for all pipeline steps. If we assume that a Monte Carlo algorithm is used for the Uncertainty Quantification, the file for example defines the number of iterations to run and which of the input parameters for the subsequent model simulation are considered to be uncertain.
The output file, regardless of chosen UQ method, always has to abide by the same format which is a list of key-value pairs. As a consequence of this prerequisite, the execution of our UQ methods will produce a series of files labelled `data.recom-i` with i the running index up to the desired sample size, where their contents only differ in the values of the specified uncertain parameters.
REcoM2 as a model is complex and running thousands of simulations would take too much time on machines that are designed for everyday work. We thus outsource this task to an external supercomputer system (see Section 4.4). The outputs of *REcoM2* are several *netCDF* (`.nc`) files that for example contain values of a biochemical variable over the course of the simulation time frame. All relevant output data is retrieved from the external system and then serves as input for the Sensitivity Analysis. Since our UQ and SA approach is independent of the underlying model and only requires some generic Quantity of Interest (QoI), we are able to circumvent a computational overhead of *REcoM2* by deploying a synthetic test model that satisfies the

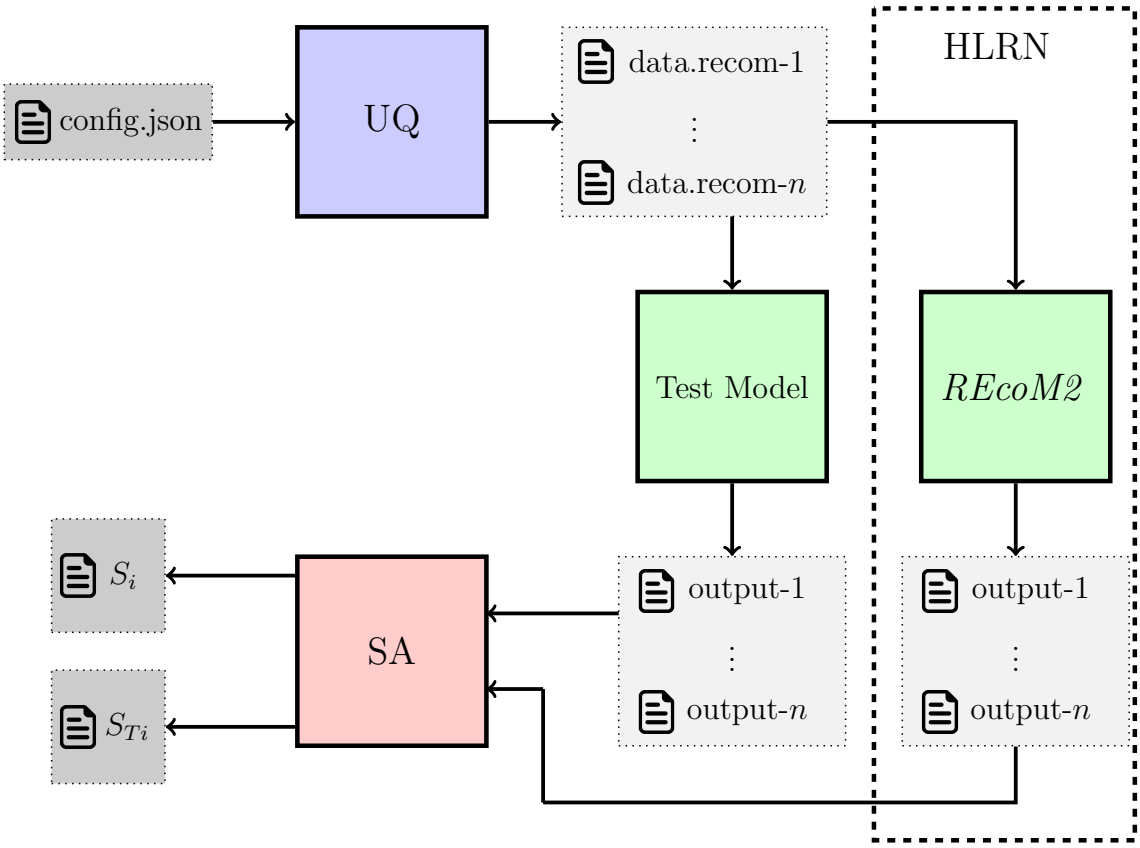


Figure 15: General layout of the pipeline to produce sensitivity indices S_i and S_{Ti}.

same input specifications as *REcoM2* (see Section 4.5).
Long story short, the UQ step produces a batch of input files with different parameter perturbations for the model and the SA step retrieves QoIs from the model output and calculates desired statistical measures.

4.2. (Quasi)-Monte Carlo

Our implementation of the (quasi)-Monte Carlo approach is based on an existing python library called *XMC* where *X* is a wildcard for different Monte Carlo variants [3]. The GitLab repository gives following description:

> "XMC is a Python library for parallel, adaptive, hierarchical Monte Carlo algorithms, aiming at reliability, modularity, extensibility and high performance."

However, we do not intend to employ a wide range of variants in this thesis and confine ourselves to the already introduced standard Monte Carlo (MC) method and quasi-Monte Carlo (qMC) method. The library was thus shrunk down to its core functionality by removing its parallelization capabilities for example. Our resulting code still retains two key qualities of *XMC*:

- A Modular implementation allows to easily integrate modifications like quasi-Monte Carlo or to exchange a functionality by replacing a specific class.
- Parameterization via settings file enables a compact and adaptable configuration of the program. Changes to its operating mode are hence possible without modifying the code base itself.

Since the model interface by design expects a series of input files (see Section 4.4), we integrated the perturbation of parameter values and generation of input files into the MC algorithm. In the following we give an overview over the steps in the (quasi)-Monte Carlo implementation.
Note that the following description is partially valid for the Stochastic Collocation method as well because MC and SC are both contained in the same UQ framework.

Initialization:
The execution is entirely configurable by a key-value pair system in the file `resources/config.json`. These settings are parsed and employed in the top level module `uq_standalone.py`. The user is further able to provide a specified set of command line arguments depicted in Table 2. Note that all arguments are optional. Besides initializing environment variables, `uq_standalone.py` assembles all configurations that are used to provide the parameters for dynamic instantiations of all modular components. This is achieved during runtime based on a file path provided in the `config.json`. We present this approach in Example 4.1.

Parameter	Description
`-m --message`	Stores an optional message in the output file `info` that allows a description of the current simulation.
`-p --path`	Specifies the output directory relative to `./out/`. Default value is the current timestamp.
`-l --level`	Overrides the logger level specified in the configuration file. Options are 0 (DEBUG), 1 (INFO), 2 (RESULT) or 3 (ERROR).

Table 2: Optional command line arguments that can be provided when executing the UQ algorithm.

Example 4.1. Values produced by a random number generator during the (quasi)-Monte Carlo execution need to be transformed onto a specified distribution. There exists a predefined superclass `TransformWrapper` from which a user defined class can inherit and provide individual implementations. Let `inverseTransformWrapper.py` be such realization containing a class `InverseTransformWrapper` which inherits from `TransformWrapper`. In order to integrate this module, it only has to be in the correct directory which in this case would be `/uq/classDefs_transformWrapper/` and needs to be addressed in the `config.json` file under the JSON node `uq/transformer/` by setting the value of key `type` to `inverseTransformWrapper.InverseTransformWrapper`.

Configurations are generally extracted by a custom method `getConfig` in the utility module `util.py` that extends straightforward dictionary parsing for example by demanding a default value and alerting the user of faulty entries.
The file `resources/data.recom` contains all relevant biochemical input parameters for *REcoM2* as key-value pairs. Since we want to keep all configurations in one place, each parameter can also be found in the configuration file under the JSON node `/parameters/` where each parameter can be linked to the `data.recom` parameter by the JSON key `recomKey`. The boolean value of subkey `uncertain` determines whether this parameter is considered in the algorithm. Note that the file `data.recom` serves only as formatting template for output generation and the values within the file function as default values on which we realize a perturbation by factor multiplication.

Main Loop:
From the entry module `uq_standalone.py` we instantiate the class `UQAlgorithm` in `uqAlgorithm.py` and run the algorithm's main execution loop. A single iteration of that loop can be boiled down to the steps of sample generation, input perturbation of the original values with these samples and a subsequent creation of input files for the biogeochemical model that incorporate the perturbed inputs. Listing 1 illustrates the modular steps of the algorithm. The number of iterations which correspond to the desired sample size can be defined in the configuration file under the JSON node `uq/uqAlgorithm/iterations`.

```
def run(sampleSize, allParameters):
  # 'allParameters' contains information about distributions
  certainParameters = filterCertain(allParameters)
  uncertainParameters = filterUncertain(allParameters)

  for i in range(sampleSize):
    # Uncertainty dimension
    dimension = len(uncertainParameters)

    # Iteration is passed for the Sobol
    # sequence offset (qMC)
    randomVector = _randomGenerator.generate(dimension, i)

    # Target distribution can be specified for
    # each uncertain parameter
    transformedVector = _transformer.transform(
            randomVector, uncertainParameters.distributions)

    # 'uncertainParameters' contains the original
    # default parameter values
    perturbedVector = _perturbator.perturb(
            uncertainParameters, transformedVector)
```

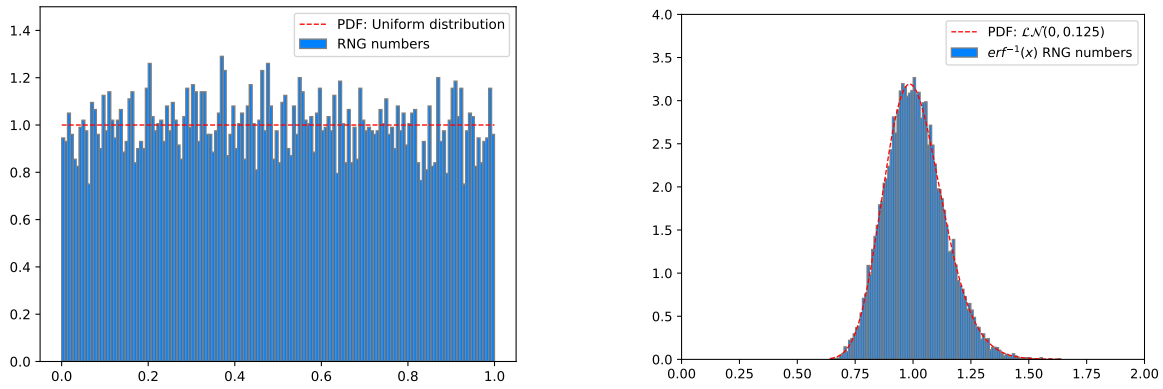
```
# Building the output file retains original values of
# 'certainParameters' and overwrites original values of
# 'uncertainParameters' with 'perturbedVector'
outputPerturbedFile(certainParameters, perturbedVector)
```

Listing 1: Main execution loop for both standard Monte Carlo and quasi-Monte Carlo where only the concrete random number generator implementation differs. Note that the algorithm does not have a return value, but outputs results directly to files.

Random Number Generation:
We need to generate (pseudo)-random values for both standard Monte Carlo and quasi-Monte Carlo. In Section 3.1.1 and Section 3.1.2 we explained that their key difference is the choice of a random number generator. For MC we use the method `random.uniform` from the standard python library `random` in order to generate uniform random values and for qMC we employ a custom module that generates quasi random values based on a Sobol Sequence. Both generators can be seen in Figure 16a and Figure 17a, respectively. Section 3.3.3 illustrated why two sample matrices are needed for the SA algorithm. The user can choose the methods for both matrices $\mathbf{A}$ and $\mathbf{B}$ in the JSON nodes `uq/uqMethodA` and `uq/uqMethodB`, respectively. As a consequence the appropriate candidate of JSON node `uq/randomGenerator/generators` is chosen accordingly to accommodate both MC variants by an option switch that is resolved during dynamic module instantiation (see Example 4.1). Note that any random number generator needs to return a (quasi)-random vector of length d where d describes the uncertainty dimension.

Transformation:
After the random number generator has returned a random vector, we call a method for each entry that transforms the value which is by default given in the inverval $[0, 1)$ onto a probability density function (see Figure 16 and Figure 17). The function can either be specified for each uncertain parameter individually in the configuration file `config.json` or can be globally overwritten by the value of JSON node `uq/transformer/pdf` if the value of `uq/transformer/overrideParameterDistribution` is set to `true`. By default we use the *inverse transform sampling* method for a transformation, but a custom approach can also be implemented (see Example 4.1).

Perturbation:
A random number generation and a subsequent transformation onto a given probability density function results in a MC sample. Accordingly, this sample is a vector of transformed random numbers. The goal of this step is applying each entry of this vector to the uncertain parameters. In our context, *applying* the sample means a factor multiplication with the original value of the uncertain parameter. Remember that the configuration file `config.json` does not actually hold the default values of the parameters, but rather contains a link to the corresponding parameter in the file `data.recom`. In our default perturbation class (which again may be replaced by a



(a) (b)

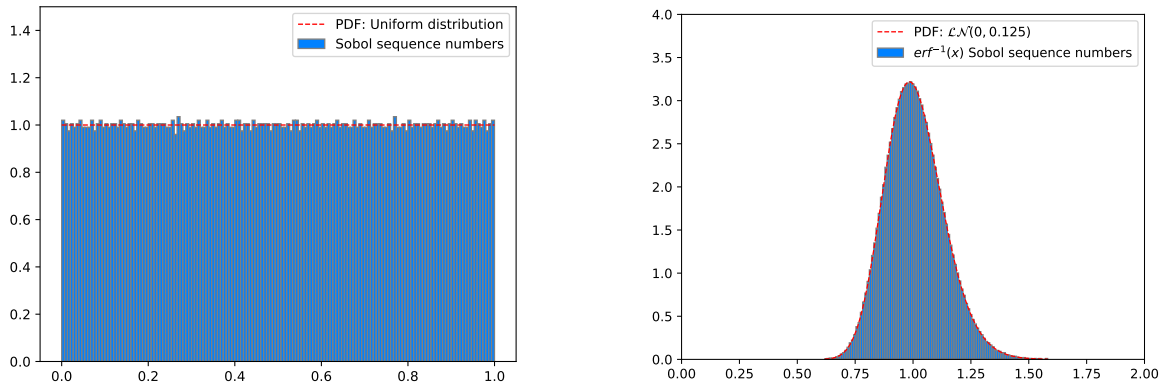
Figure 16: Inverse Transform Method with a Python random number generator. Figure (a) illustrates the relation between the pdf of a uniform distribution and the relative frequencies of the Python random number generator numbers. Figure (b) shows the frequencies of these numbers after their transformation onto a lognormal distribution. We used a sample size of 10000.

custom implementation) we therefore demand a path to the `data.recom` file under the JSON node `uq/perturbator/inputPath`. When calling a perturbation of an uncertain parameter with a sample, the value of the corresponding key in `data.recom` is loaded before multiplying it with the sample value.

Output:
The main loop of the Monte Carlo algorithm does not return some result, but outputs it directly into files at the end of each iteration. In order to map uncertain parameters to their newly perturbed values, we use tuples of the form $(recomKey, perturbedValue)$ where $recomKey$ describes the identifier of a parameter in both the configuration file `config.json` as well as the model input file `data.recom` and where $perturbedValue$ is the result from the perturbation step.
Section 3.3.1 presented common sensitivity measures. We will calculate both the normalized first order index (S_i) and the normalized total effect index (S_{Ti}) where the index i refers to the ith uncertain parameter ξ_i. Observe upon re-examining Table 1 that both measures need the total variance $\mathbb{V}[y]$ for their calculation. S_i further requires the variance of the conditional expectation value where ξ_i is considered to be certain and S_{Ti} demands the variance of the conditional expectation value where all uncertain parameters except ξ_i are supposed to be known. A naive consideration could assume that the number m of input files that later need to be executed separately by the model could be calculated by the following formula where n describes the sample size and d the number of uncertain parameters:

$$m = n \cdot (1 + 2d) \tag{48}$$

It becomes clear that the time complexity of cumulated model simulations is not only dependent on a given sample size, but is also greatly influenced by the uncertainty



(a) (b)

Figure 17: Inverse Transform Method with quasi-random numbers generated from a Sobol sequence. Figure (a) illustrates the relation between the pdf of a uniform distribution and the relative frequencies of quasi-random numbers. Figure (b) shows the frequencies of these numbers after their transformation onto a lognormal distribution. We used a sample size of 10000.

dimension. However, Equations (46) and (47) argued that only the triplet $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{A_B}^i$ needs to be evaluated by some model function f. $\mathbf{A}$ and $\mathbf{B}$ represent a sample matrix where every parameter is perturbed, respectively. $\mathbf{A_B}^i$ are the matrices for $1 \leq i \leq d$ with d the number of uncertain parameters where parameter ξ_i is assumed to be fixed. We hence improve the number of simulations from the naive consideration drastically in Equation (49) which corresponds to the matrix triplet:

$$m = n \cdot (2 + d) \tag{49}$$

Note that generating both $\mathbf{A}$ and $\mathbf{B}$ naturally requires the generation and manipulation of two independent sets of samples during prior UQ steps.
REcoM2 and also our toy model do not distinguish between an input file where, let us say, all parameters are perturbed and a file where maybe only a single one is altered. The MC algorithm thus produces files with `/resources/data.recom` as its format template that carry the same contents except for the perturbed parameter values. Since we number the MC output files consecutively, there appears to be no easy way for a Sensitivity Analysis to tell whether a model output was produced by an input file where for example only the first parameter was uncertain. We avoid any kind of confusion by a file numbering convention that is closely linked to Equation (49). Example 4.2 demonstrates this convention.

Example 4.2. Suppose a user wants to perform a Sensitivity Analysis on three uncertain parameters (ξ_1, ξ_2, ξ_3) with the aforementioned sample matrix triplet $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{A_B}^i$. The standard MC algorithm could be configured for this purpose as shown in Table 3.

At the end of the first iteration of the main loop the MC algorithm will pass the

JSON node `uq/uqAlgorithm/*`	**value**	**Description**
`iterations`	100	The used sample size (n)
`outputFullAPerturbedFile`	*true*	Whether to output files where every uncertain parameter is perturbed ($\mathbf{A}$)
`outputFullBPerturbedFile`	*true*	Whether to output files where every uncertain parameter is perturbed ($\mathbf{B}$)
`outputSingleAPerturbedFile`	*true*	Whether to output files where all but one uncertain parameter from $\mathbf{A}$ are perturbed at a time (${\mathbf{A}_\mathbf{B}}^i$)
`outputSingleBPerturbedFile`	*false*	Whether to output files where all but one uncertain parameter from $\mathbf{B}$ are perturbed at a time (${\mathbf{B}_\mathbf{A}}^i$)
`uqMethodA`	*mc*	Sampling strategy for sampling matrix $\mathbf{A}$
`uqMethodB`	*mc*	Sampling strategy for sampling matrix $\mathbf{B}$

Table 3: Exemplary configuration where the user wants to compute the sensitivities S_i and S_{Ti} in a subsequent Sensitivity Analysis with the standard Monte Carlo approach. Even though not used in this work we still allow for the user to output the files that correspond to the sample matrix ${\mathbf{B}_\mathbf{A}}^i$.

list $[(\xi_1, x_{A1}, x_{B1}), (\xi_2, x_{A2}, x_{B2}), (\xi_3, x_{A3}, x_{B3})]$ to the output function where x_{Ai} and x_{Bi} are the perturbed values for uncertain parameter ξ_i from sample matrix $\mathbf{A}$ and $\mathbf{B}$, respectively. Since `outputFullAPerturbedFile` is true, the file `data.recom-001` will be created. With a sample size of $n = 100$, the indices 001 up to 100 are reserved for all files regarding $\mathbf{A}$. Analogously, files `data.recom-101` up to `data.recom-200` are created because `outputFullBPerturbedFile` is true. The value of `outputSingleAPerturbedFile` is also true such that the files `data.recom-201`, `data.recom-202` and `data.recom-203` will be created after the first iteration. They implement the case where every uncertain parameter is perturbed with sample matrix $\mathbf{A}$ except ξ_1, ξ_2 and ξ_3, respectively, which are fixed with the corresponding column from $\mathbf{B}$. This corresponds to the first row of matrix ${\mathbf{A}_\mathbf{B}}^1$, ${\mathbf{A}_\mathbf{B}}^2$ and ${\mathbf{A}_\mathbf{B}}^3$, respectively. With a sample size of $n = 100$ and $d = 3$ uncertain parameters, the indices 201 up to 500 are reserved for these cases. If `outputSingleBPerturbedFile` was also set to true, then the files with indices 501 to 800 would have been created for the sample matrix ${\mathbf{B}_\mathbf{A}}^i$. The file indices of each file output type drop the offset if an output switch was set to false. For example, if the user intended to use the matrix triplet $\mathbf{A}$, $\mathbf{B}$ and ${\mathbf{B}_\mathbf{A}}^i$ for the SA and prevented the output of ${\mathbf{A}_\mathbf{B}}^i$ by disabling `outputSingleAPerturbedFile` then the file indices for ${\mathbf{B}_\mathbf{A}}^i$ would have been shifted by 300 accordingly to 201 up to 500.
With our configuration from Table 3 the algorithm produces 500 files which verifies Equation (49) because

$$m = n \cdot (2 + d) = 100 \cdot (2 + 3) = 500 \ .$$

The file indices are generally padded with preceding zeros such that all indices for example have an effective string length of three for $100 \leq m \leq 999$ where m is the total number of files created by the MC algorithm.
All output will be written to the base output directory `/out/`$\{id\}$ where $\{id\}$ by default is the current timestamp at the start of the algorithm or the user defined output path (see Table 2). Besides all input parameter files for the model, this directory also contains some metadata that is listed in Table 4. Note that this directory is given as input for the Sensitivity Analysis. This implies that model output will be placed in this directory as well as the output of the Sensitivity Analysis.

File in `/out/`$\{id\}$	**Description**
`data/`	Directory for all `data.recom-x` files.
`config.json`	Configuration used for this particular UQ run.
`data.recom`	Base parameter file. Values of uncertain parameters from this file were perturbed by the UQ algorithm.
`log`	All console output during execution. JSON node `/logger/writeToFile` must be `true`.
`samplesA`	Plain list of all perturbed sample values produced for each uncertain parameter for sample matrix $\mathbf{A}$. JSON node `uq/uqAlgorithm/outputVectorAPerturbedFile` must be `true`.
`samplesB`	Plain list of all perturbed sample values produced for each uncertain parameter for sample matrix $\mathbf{B}$. JSON node `uq/uqAlgorithm/outputVectorBPerturbedFile` must be `true`.
`sobolSeedA`	Contains the Sobol seed used for sample matrix $\mathbf{A}$. File only produced if it is a quasi-Monte Carlo run.
`sobolSeedB`	Contains the Sobol seed used for sample matrix $\mathbf{B}$. File only produced if it is a quasi-Monte Carlo run.
`info`	Carries the current timestamp at the start of the MC algorithm as well as an optional description from the user (see Table 2).

Table 4: List of output produced by a (quasi)-Monte Carlo simulation.

4.3. Stochastic Collocation

Last section showed the common workflow if a Monte Carlo approach is chosen as the desired UQ method. Our implementation aims to incorporate the SC method in the framework even though Stochastic Collocation will not be used to calculate sensitivity indices, but rather to approximate sample statistics (means and variance). The only modification to the existing workflow is essentially to not produce samples based on two sample matrices ($\mathbf{A}$, $\mathbf{B}$ and in turn $\mathbf{A_B}^i$ and $\mathbf{B_A}^i$), but merely output a single

one (e.g. **A**). Table 5 shows an exemplary configuration if the user wants to employ Stochastic Collocation as the UQ method.

JSON node `uq/uqAlgorithm/*`	**value**	**Description**
`outputFullAPerturbedFile`	*true*	Whether to output files where every uncertain parameter is perturbed (**A**).
`outputFullBPerturbedFile`	*false*	Whether to output files where every uncertain parameter is perturbed (**B**).
`outputSingleAPerturbedFile`	*false*	Whether to output files where all but one uncertain parameter from **A** are perturbed at a time ($\mathbf{A_B}^i$).
`outputSingleBPerturbedFile`	*false*	Whether to output files where all but one uncertain parameter from **B** are perturbed at a time ($\mathbf{B_A}^i$).
`uqMethodA`	*sc*	Sampling strategy for sampling matrix **A**.
`uqMethodB`		Sampling strategy for sampling matrix **B**.
JSON node `uq/sc/*`	**value**	**Description**.
`level`	8	Refinement level for sparse grid generation.

Table 5: Exemplary configuration where the user wants to compute expectation value and variance in a subsequent Sensitivity Analysis with Stochastic Collocation. Note that `uqMethodB` is left blank and that only files of type *FullA* will be produced.

Sparse Grid Generation:
Unlike previous MC methods, Stochastic Collocation does not create samples based on (quasi)-random numbers. It rather creates a set of collocation points (nodes) at which the underlying function has to be evaluated (see Section 3.1.3). We implemented the class `sparseGrid` that computes such nodes and corresponding weights. Note that this class creates a sparse grid with Clenshaw Curtis (CC) nodes and that the process is deterministic.
The only modifying parameter is the refinement level found in JSON node `uq/sc/level` which determines the number of nodes and subsequently the nodes' density in the d-dimensional hypercube where d is the number of uncertain parameters. A point in this hypercube corresponds to a d-dimensional (quasi)-random vector created in the (quasi)-Monte Carlo method. However, it is important to acknowledge their respective value domains. The random vector is commonly given on $[0, 1)$ whereas all CC nodes lie in $[-1, 1]$. Since a subsequent transformation onto a given probability density function (see Section 4.2) expects values distributed on former domain, we shift

all node values based on a target domain. For both sample matrices, this shift can be configured in JSON nodes `uq/randomGenerator/domainTranslationA` and `uq/randomGenerator/domainTranslationB`, respectively.

Output:
Table 4 showed the output produced for a (quasi)-Monte Carlo simulation. The list is extended by entries of Table 6 if the SC method is chosen. Note that file `weights` is essential because it serves as input for the Sensitivity Analysis where node weights are used for the calculation of expectation value and variance according to Equations (24) and (25).

File in `/out/`$\{id\}$	**Description**
`totalIterations`	Number of nodes in the sparse grid.
`nodes`	Contains all nodes of the sparse grid before transformation such that all entries are given in the domain $[-1, 1]$.
`weights`	Weights associated with the nodes in the sparse grid.

Table 6: Additional items produced by the Stochastic Collocation method.

4.4. Model Interface

REcoM2 is currently being maintained by researchers at the *Alfred-Wegener-Institut* (AWI) [1]. For functional details on *REcoM2* see Section 2. Providing insight into the technical code base of *REcoM2* is beyond the scope of this thesis. In this section we rather focus on the interface of the model by viewing it as a black box model with some input and output.
The model is dependent on several input files which on the one hand cover the initialization of boundary conditions and oceans forcings. On the other hand there exists an input file called `data.recom` that encompasses the range of all biochemical parameters. Since our Sensitivity Analysis will only consider parameters from that particular file, we are able to contain all uncertainty in one place.
Even a single simulation of the model is quite costly in terms of computational time due to complex integrations. It becomes clear that it would simply be unfeasible to execute the model on local machines if we want to obtain a reasonable sample size. For this reason we outsourced the model simulation to the distributed supercomputer system hosted at the sites *Georg-August-Universität Göttingen* and *Zuse Institute Berlin*. The alliance operating the system is the HLRN (*Norddeutscher Verbund zur Förderung des Hoch- und Höchstleistungsrechnens*) which aims at providing means for high performance computing to scientific institutions throughout Germany. Figure 18 portrays a physical section of the LISE system in Berlin. In the 56th edition of the "TOP500" the Emmy system in Göttingen even reached the 47th place, clocking in at $5.948,8 TFlop/s$ [57]. For our computations we can use up to 16 standard compute node that feature following specifications [22]:

- **2** CPUs + 1 Intel Omni-Path host fabric adapter
- Per CPU 1 Intel Cascade Lake Platinum 9242 (CLX-AP), **48** cores

Figure 18: HLRN-IV system LISE in Berlin. Photo: ITMZ | University Rostock

Note that these specifications are valid for both systems in Berlin and Göttingen and that we will not distinguish between them. With 2 CPUs and 48 cores per CPU we are therefore able to run 96 simulations in parallel. For benchmarks regarding the actual execution performance we refer to Section 5. The system employs SLURM as their batch scheduler which allows a job script based execution. A typical job submission script is displayed in Listing 2 by which 1000 simulations will be scheduled for execution.

```
1  #!/bin/bash
2  #SBATCH --job-name=recom
3  #SBATCH --time=02:00:00
4  #SBATCH --nodes=1
5  #SBATCH --ntasks=96
6  #SBATCH --mem-per-cpu=1500M
7  cat $0
8  module load intel/18.0.6
9  module load impi/2018.5
10 module load netcdf/intel/4.7.3
11 module load hdf5/intel/1.10.5
12 export NETCDF_ROOT=/sw/dataformats/netcdf/intel.18/4.7.3/skl
```

```
13 export MPI_ROOT=$(dirname $(dirname `which mpiifort`))
14 export MPI_INC_DIR=${MPI_ROOT}/include
15 export TMPDIR=/tmp
16 #
17 srun="srun --exclusive -N1 -n1"
18 #
19 parallel="parallel --delay 0.2 -j $SLURM_NTASKS \
          --joblog recom_runtask.log.$SLURM_JOB_ID --resume"
20 #
21 $parallel "mkdir $LOCAL_TMPDIR/bin-{} \
       && cp ./build/mitgcmuv $LOCAL_TMPDIR/bin-{} \
       && $srun $LOCAL_TMPDIR/bin-{}/mitgcmuv > slurm.out \
       && mv recomDiags2D.nc $HOME/recomDiags2D-{} \
       && mv recomDiags3D.nc $HOME/recomDiags3D-{} \
       && rm -r $LOCAL_TMPDIR/bin-{}" ::: {0001..1000}
22 #
23 sacct -j $SLURM_JOB_ID
           --format="JobID,NodeList,AveVMSize,MaxVMSize,
           MaxRSS,Start,CPUTime,Elapsed"
```

Listing 2: Job submission script for the SLURM environment. Note that this setup allows for 96 concurrent tasks and that we are granted 2 hours of wall time to finish 1000 simulations.

Revisit the command in line 21 of Listing 2. This line essentially tells the scheduler that it is supposed to distribute the execution of `mitgcmuv` inside the directories `$LOCAL_TMPDIR/bin-i` for i the left padded running index to the available 96 cores. The directory `$LOCAL_TMPDIR` resides in the transient memory of the compute node itself which mitigates operations on the server filesystem. Remember that biochemical parameters are located in the input file `data.recom` and that we create a batch of perturbed input files `data.recom-i` with i the left padded running index in a UQ simulation. Since each of the model instances requires its individual `data.recom` file, we link it to the respective file `data.recom-i` during the directory setup. As a consequence, the batch of perturbed parameter files from the output of a UQ method need to be copied to the HLRN file system because the Uncertainty Quantification is executed outside their system.

Once all simulations have successfully terminated, the necessary output files produced in each directory `$LOCAL_TMPDIR/bin-i` have been moved to the `$HOME` directory and may be copied to the respective output folder of the corresponding UQ run. From this point, it is the task of a Sensitivity Analysis to extract some desired quantity of interest from these output files.

4.5. Test Model

Last section revealed how expensive the *REcoM2* model is with its actual execution of the code on the one hand and the overhead of manual initialization and copying processes on the other hand. For this reason we implemented a test model that employs simple simulation functions and avoids the overhead because the program can be run on the same system as the Uncertainty Quantification and the Sensitivity Analysis.

Initialization:
Initialization is performed by the top level module `model_standalone.py` which works analogously to `uq_standalone.py` presented in Section 4.2. Unlike for the UQ simulation we demand a command line argument that serves as the base input directory for the model. It has to be some path residing in the UQ output directory `./out/`. Remember that the original configuration file `./resources/config.json` has been copied to this base directory in the state at the time of the UQ method execution. This approach implicates that the configuration of the test model (and that of a subsequent SA) are both loaded from this file and motivates the user to set up a configuration for all steps in this pipeline beforehand. Furthermore, the model even needs some configuration defined for Monte Carlo or Stochastic Collocation for its own execution (see Tables 3 and 5). Following configurations in Table 7 are meant for the test model only. Similar to Example 4.1, the variable `simulationFunction` enables the user to easily implement and integrate a custom simulation function.

JSON node `/model/*`	**Description**
`qoiFileName`	Name of the file where results of a model iteration will be written to.
`simulationFunction`	Test function used in each iteration.

Table 7: Additional parameters configurable in the file `config.json` located in the model input directory.

Main Loop:
From the entry module `model_standalone.py` we instantiate the class `ModelSimulation` in `./model/modelSimulation.py` and run the main execution loop. The first step is a file structure integrity check. With the configuration provided from the UQ step and Equation (49) we are able to test if the correct number of parameter files exist. An iteration over all input files executes the following steps where i describes the left padded iteration index:

- Load and parse the data from the `./data/data.recom-i` into a dictionary object.
- Call the simulation function with the loaded dictionary as input parameter. The result is specified as a tuple `(result, equation)` where *result* is the actual output value of the function and *equation* is a string based description that specifies the used equation for debug and logging purposes.

- Pass the raw result to a method that writes it to the file `./out/{id}/qoi/{qoiFileName}-i`.

We provide simple test functions like `squareValue` that can be expressed as $2+\xi_1^2+3*\xi_2+\xi_3^2+\xi_4+\cdots+\xi_d$ where ξ describes an uncertain parameter in the input parameters and where d is the number of uncertain parameters. Note that the test model produces a single value as output which contrasts the multidimensional output from *REcoM2* .

4.6. Sensitivity Analysis

The last part of our implementation is a Sensitivity Analysis in accordance to the methodology presented in Section 3.3. We maintain a code structure similar to the previous implementations because function modularity and configuration via JSON file remain a key feature.

Initialization:
Data produced by either *REcoM2* or our test model drives the SA. The input path is relative to the base path `./out/` and is defined by a positional command line argument when executing the top module `sa_standalone.py` (see Table 8). The settings

Parameter	Description
`#1` input	[Required] Determines the input path for the SA algorithm relative to `./out/`.
`-p --path`	[Optional] Specifies the output directory relative to `./out/#1/sa/`. Default value is the current timestamp.

Table 8: Command line arguments that can be provided when executing the Sensitivity Analysis algorithm.

file `./out/{id}/config.json` serves as configuration for the SA analogous to the test model where {id} corresponds to the required input command line argument. Apart from the information regarding the number of expected data files (see Table 3), the Sensitivity Analysis also loads the parameters. Note that these parameters do not serve any logical purpose regarding the working process of a SA, but they provide information about the uncertain parameters we have perturbed, for example their display name. If Stochastic Collocation was chosen as UQ method the algorithm further parses the weights from the file `./out/{id}/weights`.
Observe that the output of *REcoM2* and our test model fundamentally differs from each other in terms of their file name convention as well as their individual content. Such discrepancy should be allowed because our overall non-intrusive approach intends to treat the model as a *black box model* where some generic output may be produced. Our Sensitivity Analysis thus needs to dynamically interpret different types of output. A modular implementation that we advocate already allows these individual interpretations of output. However, in order to enable multiple distinct variations of a Sensitivity

Analysis in sequence and to arrange certain configurations for a user, we established a system of so-called SA *templates*. These templates can be found in the configuration file under the JSON node `/sa/templates/`. Templates defined here can be employed in the current SA run by adding them to the list of JSON node `/sa/useTemplates`. Table 9 shows the entries necessary for a template that will be explained in more detail in the following. Note that such template structure is specialized for applications with `.nc` files.

QoI Extraction:
The main loop of our SA implementation iterates over all desired templates specified in the configuration file under JSON node `/sa/useTemplates`. Extracting a QoI is a template specific process whereas calculating sensitivities and outputting results of a run is globally implemented regardless of the current template.
Table 9 introduces an exemplary template layout that aggregates all steps necessary for

JSON node `/sa/templates/*/`	**Description**
`qoiExtractor`	Implementation that defines the extraction process of data from the file. Must inherit from `qoiExtractorWrapper`.
`qoiFileName`	Name of the file from which the SA will parse data.
`qoiVariableList`	A list of keys that allows to extract multiple parts from an output file at once. For example, it enables us to read the same quantities for both types of plankton from `.nc` files.
`variableExtractionFunction`	Defines how the data is read out from *qoiFileName*.
`qoiOperatorFunction`	Applies some function in order to produce a QoI.
`multiVariableOperatorFunction`	Applies an operation to the list of QoIs produced from each variable in `qoiVariableList`.
`qoiPostProcessingFunction`	Allows to insert a post-processing step before returning the final QoI.

Table 9: Exemplary SA template entries that are needed if the default class `QoiExtractor` is used for QoI extraction. This composition is especially designed for work with netCDF files.

an extraction of a QoI as it will be performed for *REcoM2* output. Listing 3 illustrates these extraction steps.
Suppose QoIs were extracted in a previous SA run. Calculating sensitivities again for the same QoIs would require the overhead of another QoI extraction. This overhead can be circumvented if the JSON node `sa/qoiInputDirectory` is set to the output

directory of the previous SA run. Instead of extracting the data from the files again, data will then be parsed from the previous run which drastically decreases computational time.
Revisit Equation (49) and remember that the user could decide before the Uncertainty Quantification whether to produce perturbed output where every parameter is uncertain ($\mathbf{A}$ and $\mathbf{B}$) and where their individual columns are kept fixed with the corresponding column of the other matrix (${\mathbf{A}_\mathbf{B}}^i$ and ${\mathbf{B}_\mathbf{A}}^i$). Note that equivalent notations for configurations are `mcFullAPerturbed`, `mcFullBPerturbed`, `mcSingleAPerturbed` and `mcSingleBPerturbed`, respectively. If a function parameter in our code needs to distinguish between them it refers to the string constants *FullA*, *FullB*, *SingleA* and *SingleB*, respectively.

```
def extractAggregatedQoIs(matrixType):
  if matrixType == 'FullA' or matrixType == 'FullB':
    # result shape: (qoiDimension)
    return aggregateQoI(matrixType)
  else:
    # result shape: (numUncertainties, qoiDimension)
    return [aggregateQoI(matrixType, u)
        for u in range(numUncertainties)]

def aggregateQoI(matrixType, parameter=None):
  qoi = []
  # Calculates output file numbers based on
  # matrixType and parameter offset
  qoiFileNums = getFileNums(matrixType, parameter)

  for i in qoiFileNums:
    # File path to output file
    qoiPath = buildPath(i, _qoiFileName)

    for varName in _qoiVariableList:

      # How to extract values from the file
      extracedVar = _extractVariable(qoiPath, varName)

      # Generates a QoI from data
      qoiOperatedVar = _qoiOperator(extractedVar)

      # Defines how multiple variables (QoIs) are handled
      multiVarQoi= _multiVariableOperator(qoiOperatedVar)

    qoi.append(multiVarQoi)
```

```
    # Applies a post processing step
    return _qoiPostProcessing(qoi)
```

Listing 3: QoI extraction performed by the default implementation in class `QoiExtractor`. This whole process can be replaced with a custom implementation that inherits from abstract class `QoiExtractorWrapper`. Observe how the template definitions from Table 9 are incorporated.

Sensitivity Calculation:
With QoIs extracted for all model output data, we can use classic arithmetic operations to calculate expectation values and variances. The mathematical basis for the Jansen estimators that are commonly used to compute sensitivities S_i and S_{Ti} were given in Equation (46) and (47). See Figure 19 for a visual illustration of the components that lead to the final sensitivities. Observe that the estimators merely yield V_i and V_{Ti}, respectively, and that another estimator for the model variance $\mathbb{V}[y]$ is necessary for the sensitivity normalization.
All three estimators can be configured by the JSON nodes `sa/estimatorFunctionSi`, `sa/estimatorFunctionSTi` and `sa/estimatorFunctionVY`, respectively. In order to evaluate the quality of our estimations, we further allow for a calculation of confidence intervals around the results (see Section 3.3.3). The implemented bootstrap method can be configured accordingly with the JSON nodes given in Table 10.

JSON node `/sa/*`	Default	Description
`bootstrapReplicas`	10000	Number of repetitions for the subsampling process.
`subsampleFactor`	0.25	Specifies the size of one subsample set relative to the original sample size.
`confidenceInterval`	0.95	Range of the confidence interval.

Table 10: Configurations regarding the confidence interval calculation.

4.7. Academic Validation

Before starting our case study, we want to validate and test our implementation to some extent. Even though we will not verify the correctness of individual functions, the manual examination of a prepared test setting suffices to confirm the goodness of results that will be obtained in the case study.

4.7.1. UQ Model

The correctness of our implemented UQ methods can be reliably validated by their respective convergence rates presented in Section 3.1.1 that are recapitulated in Table 11. Numerically studied error patterns are shown in Figures 20 and 21. The methods

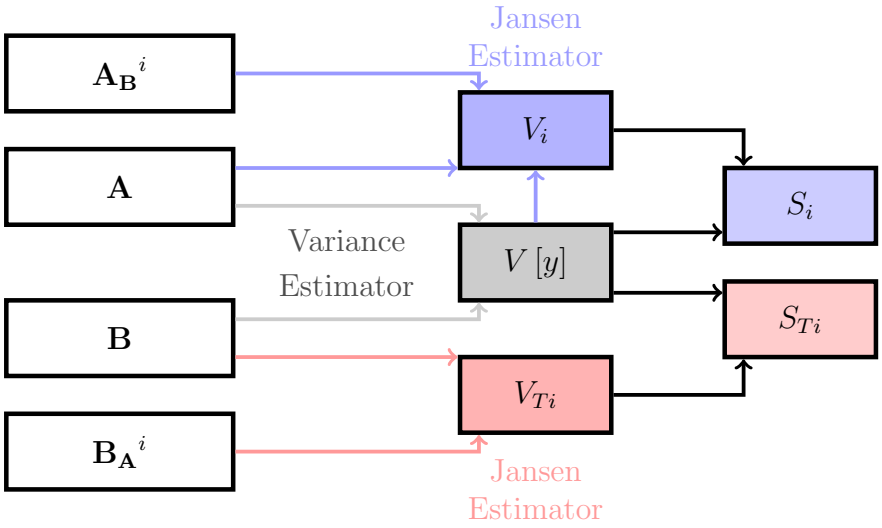


Figure 19: Components and steps involved in order to calculate the normalized first order index and the normalized total effect index. Observe that the estimator from Jansen for the calculation of V_i also requires the overall model variance $V[y]$ as input whereas the estimator for V_{Ti} does not.

Monte Carlo	quasi-Monte Carlo	Sparse Grid Stochastic Collocation
$\mathcal{O}(1/\sqrt{N})$	$\mathcal{O}((\log N)^d/N)$	$\mathcal{O}(N^{-\alpha}(\log N)^{(d-1)(\alpha+1)})$

Table 11: Order of the convergence rates of the presented UQ methods.

are compared in terms of their error in expectation and variance relative to an analytical result of a univariate polynomial function. Note that this translates to a probability space of $d = 1$ such that quasi-Monte Carlo clearly outperforms classic Monte Carlo even for a relatively small sample size even though Monte Carlo would in theory result in a total convergence for $N \to \infty$. The presented numerical results confirm our theoretical assumptions which are plotted in dashed lines. As for the sparse grid Stochastic Collocation, it is interesting to observe its convergence behaviour since it depends on the unknown smoothness α of the underlying problem. Our test case is quite simple such that it performs best with a convergence of order $\mathcal{O}(1/N^2)$, but it has to be assumed that performance could be worse in non-trivial scenarios.

4.7.2. SA Model

In order to evaluate our SA implementation we numerically retrieve the sensitivities of a function for which analytically calculated values exist. A widely used function

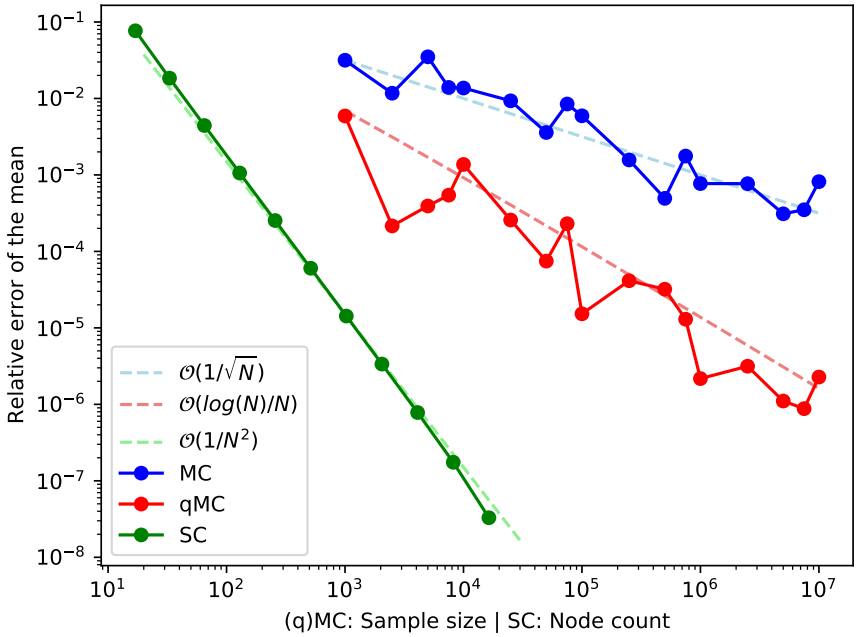


Figure 20: Convergence behaviour of the presented UQ methods in terms of their relative error in the mean with respect to an analytical solution.

is the Ishigami function that exhibits strong non-linearity and non-monotonicity [23]. Equation (50) shows the function with coefficients 7 and 0.1 used by Marrel et al. where ξ_1, ξ_2 and ξ_3 are uniformly distributed random variables on $[-\pi, \pi]$ [29].

$$f(\xi_1, \xi_2, \xi_3) = \sin \xi_1 + 7 \sin^2 \xi_2 + 0.1 \; \xi_3^4 \sin \xi_1 \tag{50}$$

Analytically calculated sensitivities are known and can be seen in Table 12. The evaluation of the function is performed by our test model and is illustrated in Figure 25. Empirical results of a variance-based Sensitivity Analysis are presented in Figure 26 and 27. Observe that the analytical sensitivity always lies within the confidence interval which indicates a robust result for the employed sample size of 50000.
Stochastic Collocation is not designed for a direct calculation of sensitivity measures. However, we are still able to compare the sample statistics (mean and variance) of the Ishigami function with the real values and with the results obtained by our (quasi)-Monte Carlo approach. Table 13 shows the results of this comparison. Observe that SC clearly outperforms both MC methods even with a fraction of necessary function evaluations.

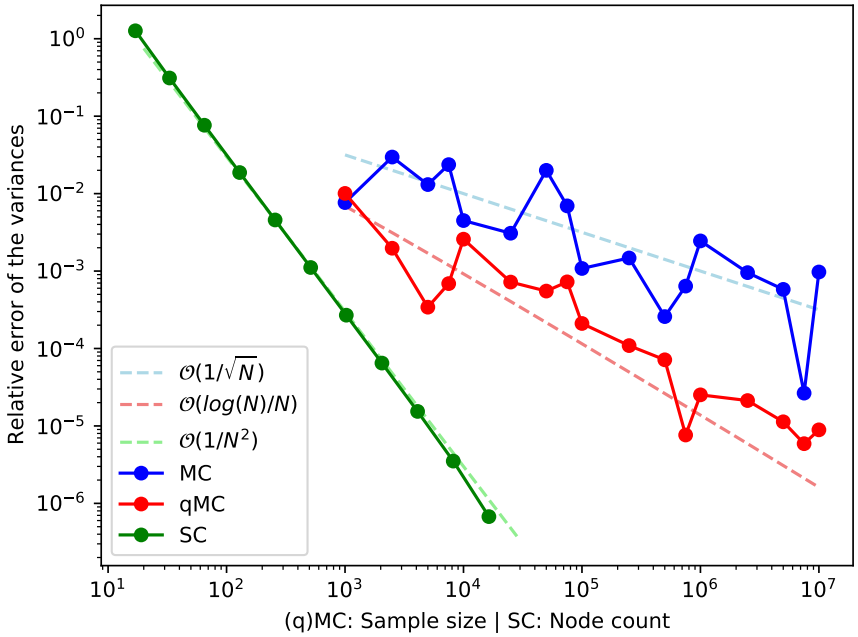


Figure 21: Convergence behaviour of the presented UQ methods in terms of their relative error in the variance with respect to an analytical solution.

Uncertainty	**sensitivity** S_i	**sensitivity** S_{Ti}
ξ_1	0.3138	0.5574
ξ_2	0.4424	0.4424
ξ_3	0.0	0.2436

Table 12: Analytically calculated sensitivities S_i and S_{Ti} for the Ishigami function.

	Exact Value	**MC**	**quasi-MC**	**SC**
Sample Size	-	10000	10000	1073 (lvl 6)
$E(Y)$	3.5	3.4245172	3.5005840	3.4999999
Error	-	7.55×10^{-2}	5.84×10^{-4}	1.8×10^{-14}
$V(Y)$	13.8445879	14.0475143	13.8398975	13.8444250
Error	-	2.02×10^{-1}	4.69×10^{-3}	1.63×10^{-4}

Table 13: Comparison between the analytical values for means and variance of the Ishigami function and results obtained by different UQ methods. Note that the sample size for SC corresponds to the node count in the underlying sparse grid.

5. Case Study

This section presents results of a variance-based Sensitivity Analysis that is performed on uncertain input parameters of the biogeochemical model *REcoM2* . All results are produced by the pipeline of algorithms introduced in Section 4. In collaboration with researchers at the *Alfred-Wegener-Institut* (AWI) we compiled Quantities of Interest (QoI) which can be seen in Table 14. These QoIs refer to data extracted during the bloom period of plankton each year.

QoI	Description
Bloom Mean Peak (BMP)	Highest value of a year averaged over five years.
Bloom Mean Value (BMV)	Average value during bloom averaged over five years.
Bloom Mean Duration (BMD)	Duration of a bloom in days averaged over five years.

Table 14: Quantities of Interest for *REcoM2* .

Activity of plankton is examined and analysed in two ways. We can either consider the net primary production (NPP) which describes how much carbon plankton take in during photosynthesis minus how much carbon they release during respiration and decay. On the other hand we can examine the concentration of chlorophyll (CHLa) that correlates with NPP because a high concentration implies a dense occurrence of phytoplankton that in turn display a generally higher cumulated productivity. Since NPP characterizes a process and a concentration of chlorophyll represents an environmental state after the production itself, it can be assumed that both quantities inherently feature a slight temporal shift.

Note that the chosen QoIs are all connected to the bloom period because this time period expresses a higher variability. A bloom commonly occurs at some time during spring season in the northern hemisphere (see Figure 22). We adapt the definition of a bloom period for a NPP (CHLa) quantity from Soppa et al. [54]: Phytoplankton bloom starts when the NPP (CHLa) value exceeds a value of 5% above the median and remains above this threshold for at least 15 days. The bloom ends after the NPP (CHLa) value drops below the threshold for 10 days. Note that the threshold periods of 15 and 10 days are derived from a multiple of 5 days which is the frequency of intermediate *REcoM2* output data. Our bloom definition thus translates to a threshold traverse of 3 and 2 consecutive model data points, respectively.

Satellites are able to measure chlorophyll on a global scale throughout the year by interpreting characteristics of light or radiance coming from the earth's surface [48]. Those measurements, however, yield reliable results only for surface water. In contrast, *REcoM2* provides CHLa values for all 30 modelled vertical water layers. The NPP quantity on the other hand is not given for each layer and is vertically integrated

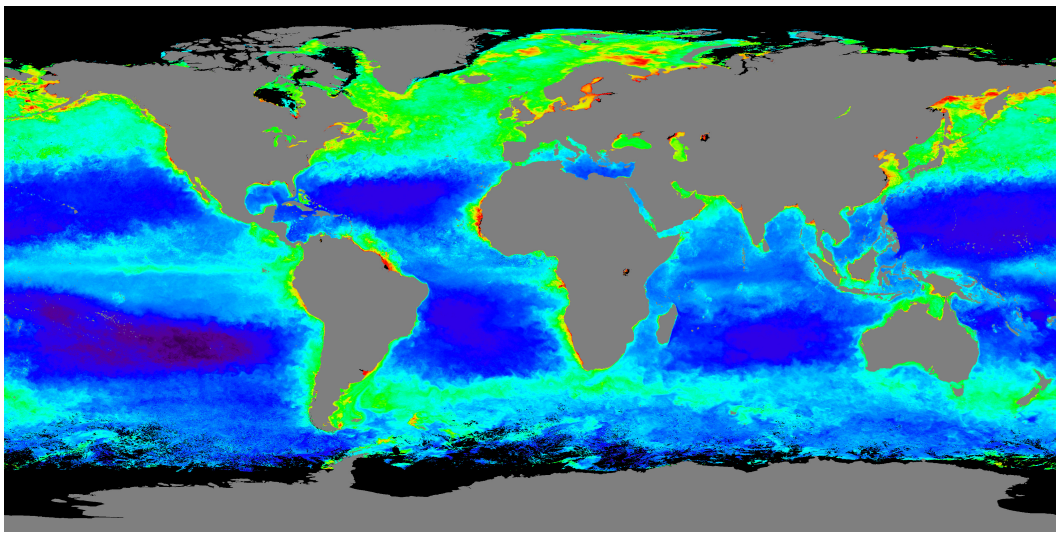


Figure 22: Chlorophyll concentration in the ocean during northern hemisphere spring season [25]. A warmer color indicates a higher concentration.

such that NPP is provided only in relation to a temporal axis whereas CHLa is further embedded in a depth dimension. In order to extract a 0-dimensional QoI from the chlorophyll data, we will examine the value of the surface layer and the value averaged over all water layers separately. This distinction is denoted as [CHLa (Surface)] and [CHLa (Average)], respectively. We hence analyse the sensitivities from Table 14 once for the NPP values and twice for the CHLa data. Figures 23 and 24 depict exemplary output of *REcoM2* . Figure 23 further illustrates the quantities that we extract from this data.

REcoM2 is configured for us such that it simulates 10 years of biochemical processes in a 1D water column in the Bermuda region where a discrete intermediate result resolution of 5 days is chosen. The model is prone to unrealistic results in the first couple of simulated years due to its initial settings and first needs to adjust itself until it reaches a balanced state. We thus omit the first 5 years entirely which is more than sufficient.

Earlier we stated that *REcoM2* is a rather complex model in terms of computational time. Since our study is confined to its one dimensional setup the complexity is in turn also quite limited. A single simulation is executed in under 400 seconds on a compute node at the HLRN (see Section 4.4). Even though this number sounds extremely low at first, one has to consider Equation (49). A large sample size in combination with several uncertain parameters could result in weeks of computation time. Fortunately, each compute node can execute 96 simulations in parallel, and we are able to reserve multiple compute nodes simultaneously depending on the overall load of the HLRN system.

The model is to be examined with regard to the sensitivity of ten parameters which are listed in Table 15. All parameters are assumed to be mutually independent. In accordance to Section 3 each parameter ξ is modelled as a lognormal distributed random

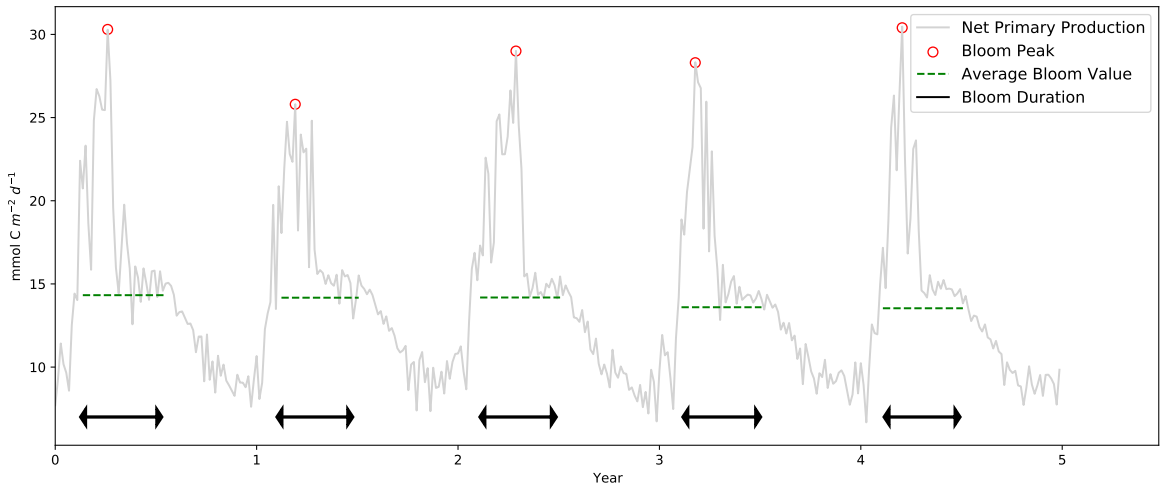


Figure 23: Cumulative net primary production of nanophytoplankton and diatoms during the last five years of *REcoM2* output. Note that the QoIs from Table 14 are 0-dimensional and that each visualized quantity must be averaged over all five years.

variable $\xi \sim \mathcal{LN}(1, 0.125)$. In Section 5.1 we present the results of a variance-based SA for all ten parameters. Results obtained by the analysis will allow us to restrict an examination to a subset of the most relevant (sensitive) parameters in Section 5.2. Fewer parameters will allow us to increase the sample size and to hopefully attain more accurate sensitivities.

5.1. High-dimensional Parameter Space

This section presents results of a variance-based SA for all ten parameters listed in Table 15 with regard to the quantity of interests shown in Table 14. Section 5.1.4 goes beyond the interpretation of individual QoIs and discusses the choice of parameters for the low-dimensional SA setup in Section 5.2. All referenced figures are attached in Appendix A.3.
An examination of ten uncertain parameters renders the quasi-Monte Carlo approach virtually infeasible because a faster convergence rate only comes into effect if the dimensionality is relatively low (see Table 11). We hence used pure Monte Carlo as a tool to produce samples. The process was performed with a sample size of 25000 such that 300000 simulations had to be executed at the HLRN according to Equation (49).

5.1.1. Bloom Mean Peak

The Bloom Mean Peak (BMP) describes the highest QoI value during bloom season averaged over the last five years of *REcoM2* output (see Figure 23). Figures 28 and 29 show the sensitivities of the BMP quantity with regard to the net primary production. Parameter alpha is clearly the most sensitive parameter. The grazing parameters

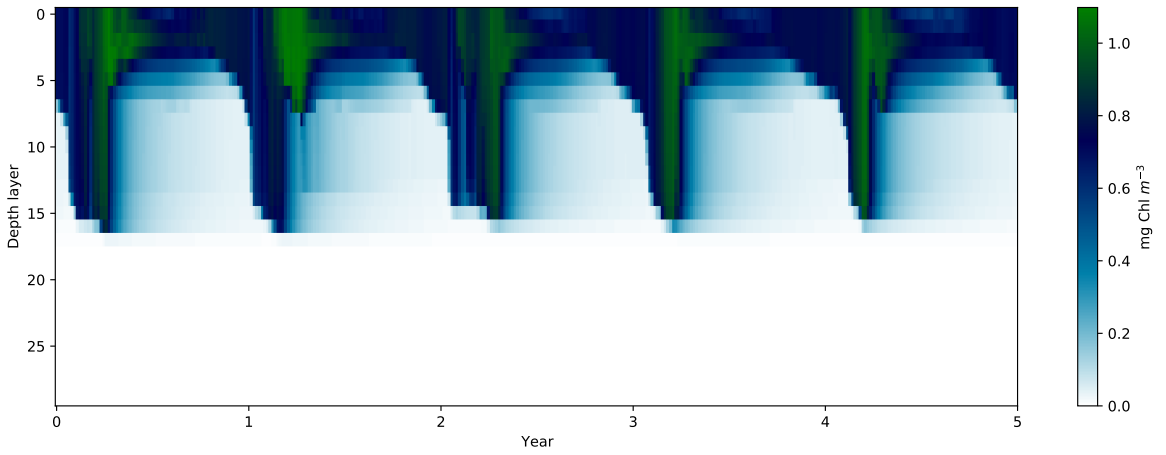


Figure 24: Chlorophyll concentration during the last five years of *REcoM2* output. Observe that QoIs from Table 14 are extracted from the surface layer as well as from the overall layer average.

graz_{max} and graz_{Eff} as well as alpha_{d} and $\text{P}_{\text{cm\_d}}$ also contribute a considerable part to the overall sensitivity. Observe that the values of the total effect indices in Figure 29 are slightly higher than their corresponding first order index in Figure 28 except for agg_{PD} which can be attributed to a numerical error. These higher values indicate that some parameters have mutual dependencies that is expressed in higher order sensitivity indices. The sum of all S_i displays the overall degree of mutual dependencies where a deviation of under 0.1 can be considered low. Remember that $\sum_i S_i = 1.0$ would imply that all variance is contained only in the first order indices and that in theory $\sum_i S_{Ti} = 1.0$ should hold as well.
Results for the BMP quantity with regard to chlorophyll concentration (CHLa) are quite similar for the surface layer and the vertical layer average which can be seen in Figures 30 to 33. Here the most influential parameters are in both cases deg_{CHL}, graz_{max} and graz_{Eff}.

5.1.2. Bloom Mean Value

Our Bloom Mean Value (BMV) quantity is defined as the mean value during bloom season averaged over the last five years of *REcoM2* output (see Figure 23). The results with regard to NPP and CHLa concentration can be seen in Figures 34 to 39 and can be interpreted analogously to the respective QoI from Section 5.1.1. However, the sum of all first order sensitivities in Figure 36 exceeds 1.0 which is mathematically not possible. Since the value is not significantly greater than 1.0 and since the confidence intervals around the individual sensitivities show a margin of error we can identify a numerical error as the cause for the excess. Note that such errors could be reduced by increasing the sample size. The CHLa (Average) data in Figures 38 and 39 show that up to six parameters may be interpreted as relatively sensitive and that there are probably little mutual interactions happening in the higher order sensitivity indices

Parameter	Default	Description
alpha $alpha_d$	0.14 0.19	Initial slope of the photosynthesis irradiance curve for both types of phytoplankton. Determines how efficient photosynthesis is at low irradiance.
P_{cm} P_{cm_d}	3.0 3.5	Maximum photosynthesis rate for both types of phytoplankton. Specifies the upper efficiency limit.
deg_{chl} deg_{chl_d}	0.1 0.1	Chlorophyll degradation rate for both types of phytoplankton.
$graz_{max}$ $graz_{Eff}$	2.4 0.4	Maximum grazing rate and grazing efficiency.
agg_{PD} agg_{PP}	0.165 0.015	Specific aggregation rate for both types of phytoplankton.

Table 15: Uncertain parameters under examination.

due to the convergence of S_i and S_{Ti} to 1.0.

5.1.3. Bloom Mean Duration

The Bloom Mean Duration (BMD) quantity is defined as the duration (in days) of the bloom season averaged over the last five years of *REcoM2* output (see Figure 23). Unlike the BMP and BMV quantities we can observe that a large portion of the overall variance can be attributed to mutual interactions of parameters. Figures 40 and 41 illustrate this phenomenon where the sums $\sum_i S_i = 0.298$ and $\sum_i S_{Ti} = 2.848$ deviate greatly from the equilibrium of 1.0. Without computing the higher order indices there is little room for the identification of sensitive parameters since every parameter has a relatively low first order sensitivity. Note that in Figure 40 the first order sensitivity of agg_{PD} is portrayed as -0.003 even though negative values are theoretically impossible. The absolute value, however, is quite low and the corresponding confidence interval includes zero as well as positive values such that we may interpret the index as zero. Even though mutual parameter interactions are lower in the CHLa QoIs (see Figures 42 to 45) the results are still not as distinct as for the BMP and BMV quantities.

5.1.4. Evaluation

Previous sections presented results of a Sensitivity Analysis where all ten parameters from Table 15 are assumed to be uncertain. In order to narrow down the considered probability space we need to exclude some of these parameters. Since first order sensitivity indices are expressive when it comes to the relative significance of parameters we employ a metric that picks out the most influential parameters based on their ranking of S_i values in the QoIs presented in the last sections. We define a parameter to be relatively sensitive if $S_i \geq \frac{1}{d} = 0.1$ for $d = 10$ the number of uncertain parameters. Table

16 shows the number of times the respective parameter was sensitive for a given QoI. Note that there are nine test cases (three for NPP and six for CHLa concentration). The highlighted rows show the most relevant parameters with regard to the examined QoIs and these parameters will thus be chosen for the shrunk dimensional space in the next section.

Parameter \ QoI	**BMP**	**BMV**	**BMD**	Σ
alpha	1	1	3	5
alpha_{d}	1	1	1	3
P_{cm}	0	0	0	0
$\text{P}_{\text{cm\_d}}$	1	0	0	1
deg_{chl}	2	2	1	5
$\text{deg}_{\text{chl\_d}}$	0	1	1	2
graz_{max}	3	3	0	6
graz_{Eff}	3	3	2	8
agg_{PD}	0	0	0	0
agg_{PP}	0	0	0	0

Table 16: Ranking of each parameter in terms of their relative sensitivity.

5.2. Low-dimensional Parameter Space

Last section presented the results of a Sensitivity Analysis for a 10-dimensional setup where we used the standard Monte Carlo approach with a sample size of 25000. Goal of this section is to confine ourselves to the restricted set of highly sensitive parameters identified in Table 16. This more focused view allows us to increase the sample size and to expand the analysis of QoI data to the quasi-Monte Carlo and Stochastic Collocation methods. All referenced figures are attached in Appendix A.4 and A.5.

5.2.1. Monte Carlo

According to Equation (49) the SA performed in Section 5.1 required 300000 model evaluations for a base sample size of 25000. With a reduced number of only 4 uncertain parameters, we were able to double the sample size while maintaining the computational load of 300000 simulations.
A selection of results is depicted in Figures 46 to 51. The results across all nine examined QoIs draw a very similar, but consistent picture: The sensitivity indices tend to rise uniformly to their counterpart in the high-dimensional setup. This behaviour is most likely due to the redistribution of proportional variance which the sensitivity indices S_i and S_{Ti} basically express. However, there appears to be no significant development in terms of changing sensitivity rankings. The given confidence intervals

are also slightly more confined which was expected by a higher sample size. It is interesting to note that the sum of the total effect indices ($\sum(S_{Ti})$) was always closer to one than the respective counterpart in the high-dimensional setup. For example, the quantity BMD [CHLa (Average)] yielded $\sum(S_{Ti}) = 1.904$ in the high-dimensional case, whereas the corresponding quantity in the low-dimensional experiment resulted in $\sum(S_{Ti}) = 1.515$ even though their respective sums of first order indices remained approximately the same. This phenomenon can probably be attributed to a generally lower number of uncertain parameters since less mutual interactions occur and thus fewer higher order indices get included in the total effect index of one parameter.

5.2.2. Quasi-Monte Carlo

According to Table 11 a better convergence rate of the quasi-Monte Carlo method with $d = 4$ as opposed to the standard MC is only given with a relatively high number of samples. We still conducted an experiment with settings analogous to those applied in Section 5.2.1, but where we used the quasi-Monte Carlo approach instead. It was expected that a sample size of 50000 should still result in fairly good result that are close to those obtained with the standard MC.
Unfortunately, the actual results revealed that the values for both S_i and S_{Ti} rendered unusable. Most of the time negative or unreasonably high values were output by the estimators. We took a step back and considered the Ishigami function from Equation (50) again. Even though we depicted only the SA results for this function based on the MC method, we also conducted an analysis for the qMC that yielded reasonable values. What we found was that the estimators are highly susceptible to the chosen seeds for the Sobol sequence wich is used in the quasi-random number generation of the qMC process. Remember that two independent sample matrices $\mathbf{A}$ and $\mathbf{B}$ are built up for the calculation of the sensitivity indices. Quasi-Monte Carlo hence generates two continuous lists of quasi-random numbers with length N from the Sobol sequence that start from specified seed numbers $seed_A$ and $seed_B$, respectively. Our first assumption was an overlap of both lists such that both sample matrices would contain a continuous sublist of duplicate entries. However, this special case of $max(seed_A, seed_B) < min(seed_A, seed_B) + N$ is not the culprit as the same effect showed up in completely separated lists.
Sadly, we were not able to identify the definite source of this behaviour for the sensitivity calculation, but it is clear that it most certainly concerns the low-discrepency of the drawn quasi-random numbers that produce unwanted interconnections between the sample matrices. Table 17 illustrates this phenomenon where we chose a setting with highly overlapping seeds. The values for identical seeds become apparent once Equations (46) and (47) are considered because their sums always yield a value of zero. It is more interesting to observe that shifting the seed for sample matrix $\mathbf{B}$ by one gives us reasonable results to a limited extent and that another shift renders the method completely useless again.

$seed_A$	$seed_B$	**Uncertainty**	**sensitivity** S_i	**sensitivity** S_{Ti}
450	450	ξ_1	1.0	0.0
		ξ_2	1.0	0.0
		ξ_3	1.0	0.0
450	451	ξ_1	0.3345	0.6304
		ξ_2	0.4526	0.3358
		ξ_3	0.0337	0.3281
450	452	ξ_1	0.0697	0.9708
		ξ_2	0.2177	0.6717
		ξ_3	-0.6432	0.2585

Table 17: Sensitivities S_i and S_{Ti} for the Ishigami function that were calculated using the quasi-Monte Carlo method with a sample size of 10000.

5.2.3. UQ Methods Convergence

Previous section dealt with the extraction of sensitivity indices for the presented QoIs. It was argued that a high dimensional examination is only viable with the standard Monte Carlo approach whereas a reduction of uncertain parameters facilitates the incorporation of the quasi-Monte Carlo method whose problems were discussed in Section 5.2.2. In this section, however, we will find that qMC performs well outside the context of sensitivity indices.
We analysed the convergence rates for both Monte Carlo variants as well as for the Stochastic Collocation regarding the statistics of mean and variance for a QoI. Results for all nine considered quantities can be seen in Figures 52 to 60. Since the sample size of the Stochastic Collocation method is derived from the node count of the underlying sparse grid we are only able to examine the discrete levels of refinement. The sample size for the (quasi)-Monte Carlo approaches were thus adjusted accordingly.
Generally, we can observe that SC performs worst and qMC performs best. One has to consider, however, that these results are highly dependent on the applied context: For example, the examined QoIs are generically constructed on the one hand and the considered uncertain parameters (and hence dimension) are not universal on the other hand.

Monte Carlo:
Computed expectation values were often volatile for a relatively small sample size. Consider for example Figure 52 where a great spike was obtained for a sample size of 1000 and that the value converged quickly with a sample size of 7500. As for the variance, Monte Carlo proved to be fairly stable even for small sample sizes. It can be concluded that a sample size of at least 10000 should be applied for the standard Monte Carlo approach.

Quasi-Monte Carlo:
The results prove that qMC is a very robust method for the computation of the presented sample statistics. Expectation value and variance yielded by quasi-Monte Carlo remained on a relatively constant level across all QoIs. Even for a small sample size, where MC struggled with regard to the expectation value, quasi-Monte Carlo results are mostly consistent with those obtained by larger sample sizes.

Stochastic Collocation:
Stochastic Collocation was certainly outperformed by the Monte Carlo variants in all conducted experiments. Especially small sample sizes (sparse grid node counts) often produced high fluctuations and deviations from values supplied by (quasi)-Monte Carlo (for example see Figure 58). Results like Figure 57 prove that SC can converge to values from the other methods tough. Still, Stochastic Collocation apparently requires a larger sample size in order to converge. Considering all results, a sample size of 50000 suffices for most of the QoIs. The nearest discrete node count equivalent for this would be a refinement level of $l = 9$ that gives us exactly 46721 nodes in a 4-dimensional sparse grid. Statistics for some quantities like BMP [NPP], BMD [CHLa (Surface)] or BMD [CHLa (Average)], however, should be computed with an even higher refinement level since no clear convergence towards (quasi)-Monte Carlo values can be observed.

6. Conclusion

In this thesis we examined the biogeochemical ocean model *REcoM2* with regard to the sensitivities of its input parameters. Since *REcoM2* is a rather complex model and its inner workings are beyond the scope of this work, it was concluded that a global Sensitivity Analysis approach would be required. We decided to aim for a variance-based Sensitivity Analysis which allowed us to treat *REcoM2* as a black-box model where only its output is of significance. Different sensitivity measures were introduced in Table 1 and the focus was set on the sensitivity indices S_i and S_{Ti}. The analysis pipeline demands a preceding Uncertainty Quantification step that evaluates the uncertainties of the considered parameters. For this task, Section 3 presented the standard Monte Carlo approach and its quasi-Monte Carlo variant as well as the Stochastic Collocation method. The implementation of the pipeline for a complete Sensitivity Analysis was introduced in Section 4.
In order to perform a variance-based Sensitivity Analysis with our implemented algorithm we defined following prerequisites in collaboration with researchers at the *Alfred-Wegener-Institut*:

- Ten parameters are considered to be uncertain (see Table 15).
- All parameters are assumed to be mutually independent.
- Uncertainty in parameters is described in the transformation of the model into a stochastic one where each uncertain parameter ξ is modelled as a lognormal distributed random variable $\xi \sim \mathcal{LN}(1, 0.125)$.
- Quantity of Interests are defined as 0-dimensional values that can be deterministically extracted from *REcoM2* output (see Table 14).

Section 5 presented results of a variance-based Sensitivity Analysis based on the standard Monte Carlo approach in accordance with the conditions above. In the high-dimensional setup with all ten uncertain parameters we concluded that the parameters like the aggregation rate for phytoplankton (agg_{PP} and agg_{PD}) are not important whereas others demonstrated a high sensitivity. These parameters are highlighted in Table 16 and they enabled us to lower the dimension to a number of four. The refined Sensitivity Analysis confirmed our previous results of the parameters' sensitivity by a higher accuracy and suggested that all four parameters are of great significance depending on the case-specific Quantity of Interest.
Quasi-Monte Carlo as an alleged improvement to the standard Monte Carlo exhibited a poor performance and rendered unusable in the context of a Sensitivity Analysis.
Stochastic Collocation cannot be directly incorporated into our Sensitivity Analysis framework to calculate sensitivities due to its deterministic sampling strategy. However, SC is known to yield precise results for sample statistics like expectation value and variance. This quality was verified with regard to the Ishigami function where SC clearly outperformed both Monte Carlo variants (see Table 13). In context of *REcoM2* , however, Stochastic Collocation could not keep up with the convergence

rates demonstrated by (quasi)-Monte Carlo where especially the quasi-Monte Carlo approach supplied robust results.

In conclusion we can declare the parameters alpha, deg_{chl}, graz_{max} and graz_{Eff} as highly sensitive and advise that emphasis is placed on them when configuring the model. The data did not allow a definite statement on the mutual dependencies of parameters because they greatly fluctuated between the examined quantities and higher order sensitivity indices were not analysed.
The Stochastic Collocation method is not viable for the calculation of expectation value or variance of *REcoM2* . Here we rather advocate the use of the quasi-Monte Carlo approach. As for the calculation of sensitivity indices, the standard Monte Carlo method should be employed because quasi-Monte Carlo failed to provide valid results. Since the calculation of the sensitivities via the Sensitivity Analysis algorithm already incorporates the computation of mean and variance it could in many cases be desirable to simply use the standard Monte Carlo method. This approach might also be favourable due to ease of implementation and sample generation speed of the standard Monte Carlo method. Furthermore, its convergence rate is theoretically not dependent on the uncertainty dimension. However, we suggest using a sample size of at least 25000 in order to obtain robust and accurate results.

6.1. Future Work

- *REcoM2* as the underlying model has been used in its 1D configuration for our thesis. For a more realistic simulation the 2D and later on the 3D setup are of great significance. Our Sensitivity Analysis could play a major role for the higher dimensional cases because a correct configuration can be deduced from sensitivities in the 1D case and the Sensitivity Analysis algorithm itself is suitable for higher dimensional setups. Of course one has to consider the massive increase in computational time of a higher dimensional model and that a Sensitivity Analysis may not be feasible for a large sample size anymore.

- We confined our examination to the first order sensitivity index S_i and total effect index S_{Ti} which in most use cases provide enough information about sensitivities. If mutual interconnections of uncertain parameters are of greater interest an extension to the higher order indices can prove useful.

- The extraction of different Quantity of Interests has been performed without a distinction between nanophytoplankton and diatoms. Defining separate analyses for each type could be interesting because the sensitivity indices may differ a lot between the individual examinations.

- In Section 3.2 we introduced the *Bayesian Method* which has not been implemented in this work. This method, however, could potentially be advantageous to the other sample based methods because a data driven approach is not entirely dependent on the premises set by the user and aims at iteratively improving itself by observation data.

- Stochastic Collocation is not used in the calculation of sensitivity indices in this work due to our implementation that is not tailored to this integration. Tang et al. among others presented a method that allows for the calculation of sensitivity indices [55].
- Monte Carlo as an Uncertainty Quantification method comes in various modifications. In this thesis we confined ourselves to the quasi-Monte Carlo variant, but an extension to other modifications are possible. For example a Multilevel Monte Carlo approach may prove beneficial for a reduction in computational time, especially once the *REcoM2* model is set up to its higher dimensional configuration.

References

[1] Alfred-Wegener-Institut. Recom2 in coupling with fesom and mitgcm, 2019. URL `https://www.awi.de/en/science/junior-groups/maresys/models.html`.

[2] G. E. B. Archer, A. Saltelli, and I. M. Sobol. Sensitivity measures,anova-like techniques and the use of bootstrap. *Journal of Statistical Computation and Simulation*, 58(2):99–120, 1997. doi: 10.1080/00949659708811825.

[3] Quentin Ayoul-Guilmard, Sundar Ganesh, Fabio Nobile, Riccardo Rossi, Riccardo Tosi, Rosa M Badia, and Ramon Amela. Xmc, 2020.

[4] Ivo Babuška, Fabio Nobile, and Raul Tempone. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 45(3):1005–1034, 2007.

[5] Jessica Blunden and Derek S Arndt. State of the climate in 2018. *Bulletin of the American Meteorological Society*, 100(9):Si–S306, 2019.

[6] Anna Borovikov, Richard Cullather, Robin Kovach, Jelena Marshak, Guillaume Vernieres, Yury Vikhliaev, Bin Zhao, and Zhao Li. Geos-5 seasonal forecast system. *Climate Dynamics*, 53(12):7335–7361, 2019.

[7] Erik T Buitenhuis, Corinne Le Quéré, and Olivier Aumont. The dynamic green ocean model: 6 plankton functional groups in an ocean global circulation model. In *JGOGS workshop, Ispra., 2002*, 2002.

[8] Sebastian Burhenne. *Monte Carlo Based Uncertainty and Sensitivity Analysis for Building Performance Simulation*. Shaker Verlag, 2013.

[9] Russel E Caflisch et al. Monte carlo and quasi-monte carlo methods. *Acta numerica*, 1998:1–49, 1998.

[10] Francesca Campolongo, Andrea Saltelli, and Jessica Cariboni. From screening to quantitative sensitivity analysis. a unified approach. *Computer Physics Communications*, 182(4):978–988, 2011.

[11] National Research Council et al. *Surface temperature reconstructions for the last 2,000 years*. National Academies Press, 2007.

[12] David C Cox and Paul Baybutt. Methods for uncertainty analysis: a comparative survey. *Risk Analysis*, 1(4):251–258, 1981.

[13] Thomas J Crowley. Causes of climate change over the past 1000 years. *Science*, 289(5477):270–277, 2000.

[14] A. R. Curtis and P. Rabinowitz. On the gaussian integration of chebyshev polynomials. *Mathematics of Computation*, 26(117):207–211, 1972. ISSN 00255718, 10886842. URL `http://www.jstor.org/stable/2004730`.

[15] Sten De Wit and Godfried Augenbroe. Analysis of uncertainty in building design evaluations and its implications. *Energy and buildings*, 34(9):951–958, 2002.

[16] Virginia A Elrod, William M Berelson, Kenneth H Coale, and Kenneth S Johnson. The flux of iron from continental shelf sediments: A missing source for global budgets. *Geophysical Research Letters*, 31(12), 2004.

[17] Michael JR Fasham, Hugh W Ducklow, and Stuart M McKelvie. A nitrogen-based model of plankton dynamics in the oceanic mixed layer. *Journal of Marine Research*, 48(3):591–639, 1990.

[18] Scientists for Future. Wir liefern die fakten. zeit zu handeln. URL `https://www.scientists4future.org/infomaterial/infomaterialien-grafiken/`.

[19] Python Software Foundation. random — generate pseudo-random numbers. URL `https://docs.python.org/3/library/random.html`.

[20] Jacques Hadamard. Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton university bulletin*, pages 49–52, 1902.

[21] Ed Hawkins and Rowan Sutton. The potential to narrow uncertainty in regional climate predictions. *Bulletin of the American Meteorological Society*, 90(8):1095–1108, 2009.

[22] HLRN. Hlrn-iv system - standard compute node - lise at zib, 2020. URL `https://www.hlrn.de/supercomputer-e/hlrn-iv-system/?lang=en`.

[23] Tsutomu Ishigami and Toshimitsu Homma. An importance quantification technique in uncertainty analysis for computer models. In *[1990] Proceedings. First International Symposium on Uncertainty Modeling and Analysis*, pages 398–403. IEEE, 1990.

[24] Michiel JW Jansen. Analysis of variance designs for model output. *Computer Physics Communications*, (1-2):35–43, 1999.

[25] Earth Observatory Jesse Allen. A world of chlorophyll, 2006. URL `https://eoimages.gsfc.nasa.gov/images/imagerecords/6000/6735/global_amosea_2006080.jpg`.

[26] Pradhan H. K., Völker C., Bracher A., Losa S. N., and Nerger L. Assimilation of global total chlorophyll oc-cci data and its impact on individual phytoplankton fields. volume 124, pages 470–490, 2019. doi: 10.1029/2018JC014329.

[27] Zouhair Lachkar, James C Orr, and Jean-Claude Dutay. Seasonal and mesoscale variability of oceanic transport of anthropogenic co2. *Biogeosciences*, 6(11):2509–2523, 2009.

[28] Iain Alexander Macdonald. *Quantifying the effects of uncertainty in building simulation*. PhD thesis, University of Strathclyde Glasgow, 2002.

[29] Amandine Marrel, Bertrand Iooss, Beatrice Laurent, and Olivier Roustant. Calculations of sobol indices for the gaussian process metamodel. *Reliability Engineering & System Safety*, 94(3):742–751, 2009.

[30] Caren Marzban. Variance-based sensitivity analysis: An illustration on the lorenz'63 model. *Monthly weather review*, 141(11):4069–4079, 2013.

[31] Lionel Mathelin and M Yousuff Hussaini. A stochastic collocation algorithm for uncertainty analysis. 2003.

[32] Colin P Morice, John J Kennedy, Nick A Rayner, and Phil D Jones. Quantifying uncertainties in global and regional temperature change using an ensemble of observational estimates: The hadcrut4 data set. *Journal of Geophysical Research: Atmospheres*, 117(D8), 2012.

[33] William J. Morokoff and Russel E. Caflisch. Quasi-monte carlo integration. *Journal of Computational Physics*, 122(2):218 – 230, 1995. ISSN 0021-9991. doi: https://doi.org/10.1006/jcph.1995.1209. URL `http://www.sciencedirect.com/science/article/pii/S0021999185712090`.

[34] Richard B Neale, Chih-Chieh Chen, Andrew Gettelman, Peter H Lauritzen, Sungsu Park, David L Williamson, Andrew J Conley, Rolando Garcia, Doug Kinnison, Jean-Francois Lamarque, et al. Description of the ncar community atmosphere model (cam 5.0). *NCAR Tech. Note NCAR/TN-486+ STR*, 1(1): 1–12, 2010.

[35] Jeremy E Oakley and Anthony O'Hagan. Probabilistic sensitivity analysis of complex models: a bayesian approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66(3):751–769, 2004.

[36] The Intergovernmental Panel on Climate Change. Ipcc temperature reconstruction ar4 wg1. URL `http://ossfoundation.us/projects/environment/global-warming/myths/images/temperature-records/2000yrs_models_ipcc_6_1_large.jpg`.

[37] J Orr, R Najjar, CL Sabine, and F Joos. Abiotic-howto. *Internal OCMIP Report, LSCE/CEA Saclay, Gifsur-Yvette, France*, 1999.

[38] Andreas Oschlies. On the use of data assimilation in biogeochemical modelling. In *Ocean Weather Forecasting*, pages 525–547. Springer, 2006.

[39] Norman A Phillips. The general circulation of the atmosphere: A numerical experiment. *Quarterly Journal of the Royal Meteorological Society*, 82(352):123–164, 1956.

[40] Dragan Poljak, Silvestar Šesnić, Mario Cvetković, Anna Šušnjara, Hrvoje Dodig, Sébastien Lalléchère, and Khalil El Khamlichi Drissi. Stochastic collocation applications in computational electromagnetics. *Mathematical Problems in Engineering*, 2018, 2018.

[41] J Rao and F Haghighat. A procedure for sensitivity analysis of airflow in multizone buildings. *Building and Environment*, 28(1):53–62, 1993.

[42] Alfred Clarence Redfield. On the proportions of organic derivatives in sea water and their relation to the composition of plankton. *James Johnstone memorial volume*, pages 176–192, 1934.

[43] Christopher L Sabine, Richard A Feely, Nicolas Gruber, Robert M Key, Kitack Lee, John L Bullister, Rik Wanninkhof, CSL Wong, Douglas WR Wallace, Bronte Tilbrook, et al. The oceanic sink for anthropogenic co2. *science*, 305(5682):367–371, 2004.

[44] Andrea Saltelli and Paola Annoni. How to avoid a perfunctory sensitivity analysis. *Environmental Modelling & Software*, 25(12):1508–1517, 2010.

[45] Andrea Saltelli and Stefano Tarantola. On the relative importance of input factors in mathematical models: safety assessment for nuclear waste disposal. *Journal of the American Statistical Association*, 97(459):702–709, 2002.

[46] Andrea Saltelli, Paola Annoni, Ivano Azzini, Francesca Campolongo, Marco Ratto, and Stefano Tarantola. Variance based sensitivity analysis of model output. design and estimator for the total sensitivity index. *Computer physics communications*, 181(2):259–270, 2010.

[47] Benjamin D Santer, KE Taylor, TML Wigley, TC Johns, PD Jones, DJ Karoly, JFB Mitchell, AH Oort, JE Penner, V Ramaswamy, et al. A search for human influences on the thermal structure of the atmosphere. *Nature*, 382(6586):39–46, 1996.

[48] Shubha Sathyendranath, Robert J.W. Brewin, Carsten Brockmann, Vanda Brotas, Ben Calton, Andrei Chuprin, Paolo Cipollini, André B. Couto, James Dingle, Roland Doerffer, Craig Donlon, Mark Dowell, Alex Farman, Mike Grant, Steve Groom, Andrew Horseman, Thomas Jackson, Hajo Krasemann, Samantha Lavender, Victor Martinez-Vicente, Constant Mazeran, Frédéric Mélin, Timothy S. Moore, Dagmar Müller, Peter Regner, Shovonlal Roy, Chris J. Steele, François Steinmetz, John Swinton, Malcolm Taberner, Adam Thompson, André Valente, Marco Zühlke, Vittorio E. Brando, Hui Feng, Gene Feldman, Bryan A. Franz, Robert Frouin, Richard W. Gould, Stanford B. Hooker, Mati Kahru, Susanne Kratzer, B. Greg Mitchell, Frank E. Muller-Karger, Heidi M. Sosik, Kenneth J. Voss, Jeremy Werdell, and Trevor Platt. An ocean-colour time series for use in climate studies: The experience of the ocean-colour climate change initiative

(oc-cci). *Sensors*, 19(19), 2019. ISSN 1424-8220. doi: 10.3390/s19194285. URL https://www.mdpi.com/1424-8220/19/19/4285.

[49] Vibe Schourup-Kristensen, Dmitry Sidorenko, Dieter A Wolf-Gladrow, and Christoph Völker. A skill assessment of the biogeochemical model recom2 coupled to the finite element sea ice–ocean model (fesom 1.3). *Geoscientific Model Development*, 7(6):2769–2802, 2014.

[50] Ralph C Smith. *Uncertainty quantification: theory, implementation, and applications*, volume 12. Siam, 2013.

[51] Sergei Abramovich Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. In *Doklady Akademii Nauk*, volume 148, pages 1042–1045. Russian Academy of Sciences, 1963.

[52] Ilya Sobol. Sensitivity estimates. 1993.

[53] IM Soboĺ. Quasi-monte carlo methods. *Progress in Nuclear Energy*, 24(1-3):55–61, 1990.

[54] Mariana A. Soppa, Christoph Völker, and Astrid Bracher. Diatom phenology in the southern ocean: Mean patterns, trends and the role of climate oscillations. *Remote Sensing*, 8(5), 2016. ISSN 2072-4292. doi: 10.3390/rs8050420. URL https://www.mdpi.com/2072-4292/8/5/420.

[55] Gary Tang, MS Eldred, and Laura P Swiler. Global sensitivity analysis for stochastic collocation expansion. *CSRI Summer Proceedings*, 2009:100, 2010.

[56] Andrei Nikolaevich Tikhonov. On the solution of ill-posed problems and the method of regularization. In *Doklady Akademii Nauk*, volume 151, pages 501–504. Russian Academy of Sciences, 1963.

[57] TOP500. Emmy+ - bullsequana x, intel xeon platinum 9242/gold 6148, intel omni-path, 2020. URL https://www.top500.org/system/179883/.

[58] Hans-Werner van Wyk, Max Gunzburger, and John Burkardt. Clenshaw-curtis type rules for statistical integrals. Technical report, Tech. rep., Florida State University, 2016.

[59] Tyler Volk and Martin I Hoffert. Ocean carbon pumps: Analysis of relative strengths and efficiencies in ocean-driven atmospheric co2 changes. *The carbon cycle and atmospheric CO2: natural variations Archean to present*, 32:99–110, 1985.

[60] Rik Wanninkhof. Relationship between wind speed and gas exchange over the ocean. *Journal of Geophysical Research: Oceans*, 97(C5):7373–7382, 1992.

[61] Neil A Weiss. *A course in probability*. Addison-Wesley, 2006.

[62] Dongbin Xiu. Fast numerical methods for stochastic computations: a review. *Communications in computational physics*, 5(2-4):242–272, 2009.

[63] Yasuhiro Yamanaka and Eiichi Tajika. The role of the vertical fluxes of particulate organic matter and calcite in the oceanic carbon cycle: Studies using an ocean biogeochemical general circulation model. *Global Biogeochemical Cycles*, 10(2): 361–382, 1996.

[64] J Zhao and C Tiede. Using a variance-based sensitivity analysis for analyzing the relation between measurements and unknown parameters of a physical model. *Nonlinear Processes in Geophysics*, 18(3):269, 2011.

A. Appendix

A.1. REcoM2 Tables

Tables 18-21 contain model and state variables for *REcoM2* . Tables 22-27 show model parameters together with their default settings specified in [49]. Table 28 explains the effects of the limiter function shown in Equation (7).

Variable	Unit	Description
DIN	mmol N m^{-3}	Dissolved inorganic nitrogen
DSi	mmol Si m^{-3}	Dissolved inorganic silicon
DFe	mmol Fe m^{-3}	Dissolved inorganic iron
DIC	mmol C m^{-3}	Dissolved inorganic carbon
Alk	mmol C m^{-3}	Alkalinity
$PhyN_{nano}$	mmol N m^{-3}	Intracellular nitrogen concentration in nanophytoplankton
$PhyC_{nano}$	mmol C m^{-3}	Intracellular carbon concentration in nanophytoplankton
PhyCalc	mmol $CaCO_3$ m^{-3}	Intracellular calcite concentration in nanophytoplankton
$PhyChl_{nano}$	mg Chl m^{-3}	Intracellular CHLa concentration in nanophytoplankton
$PhyN_{dia}$	mmol N m^{-3}	Intracellular nitrogen concentration in diatoms
$PhyC_{dia}$	mmol C m^{-3}	Intracellular carbon concentration in diatoms
PhySi	mmol Si m^{-3}	Intracellular silicon concentration in diatoms
$PhyChl_{dia}$	mg Chl m^{-3}	Intracellular CHLa concentration in diatoms
ZooN	mmol N m^{-3}	Zooplankton nitrogen concentration
ZooC	mmol C m^{-3}	Zooplankton carbon concentration
DetN	mmol N m^{-3}	Detritus nitrogen concentration
DetC	mmol C m^{-3}	Detritus carbon concentration
DetCalc	mmol $CaCo_3$ m^{-3}	Detritus calcite concentration
DetSi	mmol Si m^{-3}	Detritus silicon concentration
DON	mmol N m^{-3}	Extracellular dissolved organic nitrogen
DOC	mmol C m^{-3}	Extracellular dissolved organic carbon

Table 18: Ocean State variables

Variable	Unit	Description
BenthosN	mmol N m^{-2}	Benthos nitrogen concentration (vertically integrated)
BenthosC	mmol C m^{-2}	Benthos carbon concentration (vertically integrated)
BenthosSi	mmol Si m^{-2}	Benthos silicon concentration (vertically integrated)
BenthosCalc	mmol $CaCo_3$ m^{-2}	Benthos calcite concentration (vertically integrated)

Table 19: Benthos State variables

Variable	Unit	Description
$BenF_{Alk}$	mmol m^{-2} day^{-1}	Flux of alkalinity from benthos to bottom water
$BenF_{DIC}$	mmol C m^{-2} day^{-1}	Flux of carbon from benthos to bottom water
$BenF_{DIN}$	mmol N m^{-2} day^{-1}	Flux of nitrogen from benthos to bottom water
$BenF_{DSi}$	mmol Si m^{-2} day^{-1}	Flux of silicon from benthos to bottom water
$BenF_{DFe}$	μmol Fe m^{-2} day^{-1}	Flux of nitrogen from benthos to bottom water
$BenF_{DetCalc}$	mmol $CaCo_3$ m^{-2} day^{-1}	Flux of detritus calcite from water to benthos
$BenF_{DetC}$	mmol C m^{-2} day^{-1}	Flux of detritus carbon from water to benthos
$BenF_{DetN}$	mmol N m^{-2} day^{-1}	Flux of detritus nitrogen from water to benthos
$BenF_{DetSi}$	mmol Si m^{-2} day^{-1}	Flux of detritus silicon from water to benthos

Table 20: Benthos Variables

Variable	Unit	Description
Agg	day^{-1}	Aggregation rate
$Diss_{Calc}$	day^{-1}	Rate of calcium carbonate dissolution
Fe'	μmol Fe m^{-3}	Concentration of free iron
f_T	-	Temperature dependence of rates
G'	mmol N m^{-3}	Phytoplankton available for food intake
G_{tot}	mmol N m^{-3} day^{-1}	Total zooplankton grazing rate
G_{nano}	mmol N m^{-3} day^{-1}	Zooplankton grazing rate for nanophytoplankton
G_{dia}	mmol N m^{-3} day^{-1}	Zooplankton grazing rate for diatoms
PAR	W m^{-2}	Photosynthetically available radiation
P_{nano}, P_{dia}	day^{-1}	Carbon-specific actual rate of photosynthesis
P_{max}	day^{-1}	Carbon-specific light saturated rate of photosynthesis
r_{nano}, r_{dia}	day^{-1}	Phytoplankton respiration rate
r_{zoo}	day^{-1}	Zooplankton respiration rate
ρ^T_{Si}	day^{-1}	Temperature dependent remineralization rate of silicon
S^{chl}_{nano}, S^{chl}_{dia}	mg Chl mmol C^{-1} day^{-1}	Rate of chlorophyll synthesis
T	K	Local temperature
V^N_{nano}, V^N_{dia}	mmol N mmol C^{-1} day^{-1}	Nitrogen assimilation rate for phytoplankton
V^{Si}	mmol Si mmol C^{-1} day^{-1}	Diatom silicon assimilation rate
w_{det}	m day^{-1}	Sinking velocity of detritus

Table 21: Model Variables

Variable	Value	Unit	Description
$\epsilon_{\mathrm{N}}^{\mathrm{phy}}$	0.05	day^{-1}	Phytoplankton excretion of organic nitrogen
$\epsilon_{\mathrm{C}}^{\mathrm{phy}}$	0.1	day^{-1}	Phytoplankton excretion of organic carbon
$\epsilon_{\mathrm{N}}^{\mathrm{zoo}}$	0.1	day^{-1}	Zooplankton excretion of organic nitrogen
$\epsilon_{\mathrm{C}}^{\mathrm{zoo}}$	0.1	day^{-1}	Zooplankton excretion of organic carbon
$\rho_{\mathrm{N}}^{\mathrm{ben}}$	0.005	day^{-1}	Remineralization rate for benthos nitrogen
$\rho_{\mathrm{Si}}^{\mathrm{ben}}$	0.005	day^{-1}	Remineralization rate for benthos silicon
$\rho_{\mathrm{C}}^{\mathrm{ben}}$	0.005	day^{-1}	Remineralization rate for benthos carbon
ρ_{N}	0.11	day^{-1}	Temperature dependent remineralization of DON
ρ_{C}	0.1	day^{-1}	Temperature dependent remineralization of DOC
ρ_{Si}	0.02	day^{-1}	Temperature dependent remineralization of DetSi
ρ_{DetN}	0.165	day^{-1}	Temperature dependent degradation of DetN
ρ_{DetC}	0.15	day^{-1}	Temperature dependent degradation of DetC
$\mathrm{deg}_{\mathrm{Chl}}$	0.3	day^{-1}	Chlorophyll degradation rate

Table 22: Degradation parameters for sources minus sinks equations

Variable	Value	Unit	Description
$q^{\mathrm{Fe:N}}$	0.0008	μmol Fe mmol N^{-1}	Intracellular $Fe:N$ ratio
K_{FeL}	100.0	m^{-3} μmol	Iron stability constant
L_{T}	1.0	μmol m^{-3}	Total ligand concentration
κ_{Fe}	0.0312	m^3 mmol C^{-1} day^{-1}	Scavenging rate of iron
$q_{\mathrm{sed}}^{\mathrm{Fe:C}}$	0.011	μmol Fe mmol C^{-1}	$Fe:C$ ratio for remineralization of iron from benthos

Table 23: Parameters for iron calculations

Variable	Value	Unit	Description
ψ	0.1	-	Calcite production ratio
γ	0.3	-	Fraction of grazing flux to zooplankton pool
m_{zoo}	0.05	m^3 mmol N^{-1} day^{-1}	Zooplankton mortality rate
ϕ_{phy}	0.02	m^3 mmol N^{-1} day^{-1}	Maximal aggregation loss parameter for phytoplankton nitrogen
ϕ_{det}	0.22	m^3 mmol N^{-1} day^{-1}	Maximal aggregation loss parameter for detritus nitrogen
w_0	20.0	m day^{-1}	Detritus sinking speed at surface

Table 24: Parameters for sources minus sinks equations.

Variable	Value	Unit	Description
α_{nano}	0.19	mmol C m^2 (mg Chl W day)$^{-1}$	Light harvesting efficiency for nanophytoplankton
α_{dia}	0.23	mmol C m^2 (mg Chl W day)$^{-1}$	Light harvesting efficiency for diatoms
$K_{\text{N}}^{\text{nano}}$	0.55	mmol N m^{-3}	Half-saturation constant for nanophytoplankton nitrogen uptake
$K_{\text{N}}^{\text{dia}}$	1.00	mmol N m^{-3}	Half-saturation constant for diatom nitrogen uptake
K_{Si}	4.00	mmol Si m^{-3}	Half-saturation constant for diatom silicon uptake
$\mu_{\text{C}}^{\text{max}}$	3.00	day^{-1}	Rate of carbon-specific photosynthesis
$q_{\text{max}}^{\text{Chl:N}}$	4.20	mg Chl mmol N^{-1}	Maximum $Chl : N$ ratio for phytoplankton
res	0.01	day^{-1}	Maintenance respiration rate constant
$\sigma_{\text{N:C}}$	0.20	mmol N mmol C^{-1}	Maximum uptake ratio $N : C$
$\sigma_{\text{Si:C}}$	0.20	mmol Si mmol C^{-1}	Maximum uptake ratio $Si : C$
τ	0.01	day^{-1}	Timescale for zooplankton respiration
V_{cm}	0.70	-	Scaling factor for carbon-specific nitrogen uptake
ζ	2.33	mmol C mmol N^{-1}	Cost of biosynthesis of nitrogen

Table 25: Parameters for phytoplankton growth.

Variable	Value	Unit	Description
$f_{\text{Z}}^{\text{dia}}$	0.50	-	Relative grazing preference for diatoms
G_{max}	2.40	day^{-1}	Maximum grazing rate at 0°
K_{G}	0.35	(mmol N m^{-3})^2	Half-saturation constant for grazing loss

Table 26: Parameters for grazing.

Variable	Value	Unit	Description
$K_{\mathrm{Fe}}^{\mathrm{nano}}$	0.04	μmol Fe m^{-3}	Half-saturation constant for nanophytoplankton iron uptake
$K_{\mathrm{Fe}}^{\mathrm{dia}}$	0.12	μmol Fe m^{-3}	Half-saturation constant for diatom iron uptake
$q^{\mathrm{N:Cmin}}$	0.04	mmol N mmol C^{-1}	Min intracellular $N:C$ ratio for nanophytoplankton
$q^{\mathrm{N:Cmax}}$	0.20	mmol N mmol C^{-1}	Max intracellular $N:C$ ratio for nanophytoplankton
$q^{\mathrm{Si:Cmin}}$	0.04	mmol Si mmol C^{-1}	Min intracellular $Si:C$ ratio for diatoms
$q^{\mathrm{Si:Cmax}}$	0.80	mmol Si mmol C^{-1}	Max intracellular $Si:C$ ratio for diatoms
$s_{\mathrm{min}}^{\mathrm{N}}$	50	mmol C mmol N^{-1}	Minimum limiter regulator for nitrogen
$s_{\mathrm{max}}^{\mathrm{N}}$	1000	mmol C mmol N^{-1}	Maximum limiter regulator for nitrogen
$s_{\mathrm{min}}^{\mathrm{Si}}$	1000	mmol C mmol N^{-1}	Minimum limiter regulator for silicon
$s_{\mathrm{max}}^{\mathrm{Si}}$	1000	mmol C mmol N^{-1}	Maximum limiter regulator for silicon
T_{ref}	288.15	K	Reference temperature for the Arrhenius function

Table 27: Parameters for limitation functions.

Process	Effect of $q^{\mathbf{N:C}} \rightarrow q_{\mathbf{N:Cmax}}$
Nitrogen assimilation	Ends uptake of nitrogen
Silicon assimilation	Ends uptake of silicon
Respiration by phytoplankton	Ends release of carbon
Phytoplankton DOC excretion	Ends release of carbon
Phytoplankton DON excretion	Ends release of nitrogen
Phytoplankton calcite excretion	Ends release of carbon

Table 28: Processes modulated by the limiter function $f_{\mathrm{lim}}^{\mathrm{N:Cmax}}$.

A.2. Ishigami Plots

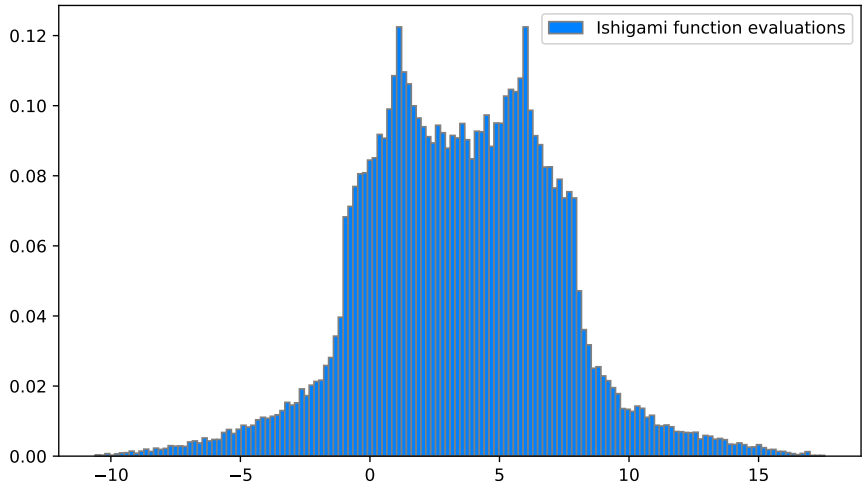


Figure 25: The Ishigami function with parameters $a = 7$ and $b = 0.1$. We used a sample size of 50000.

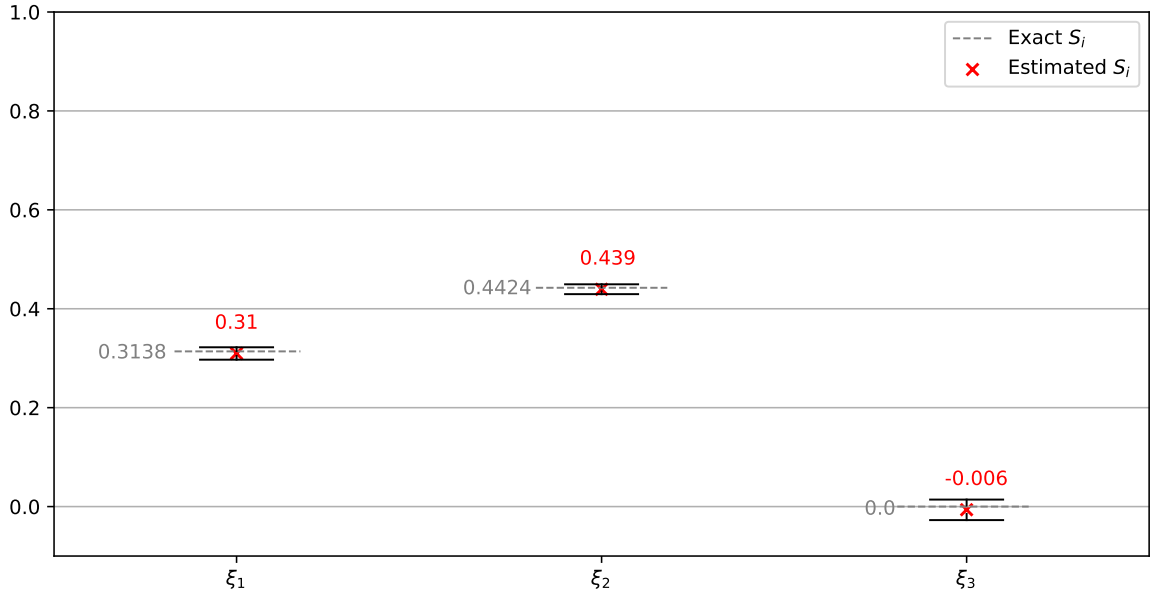


Figure 26: First order sensitivities S_i for the Ishigami function with 95% confidence intervals. We used a sample size of 50000 with the MC method.

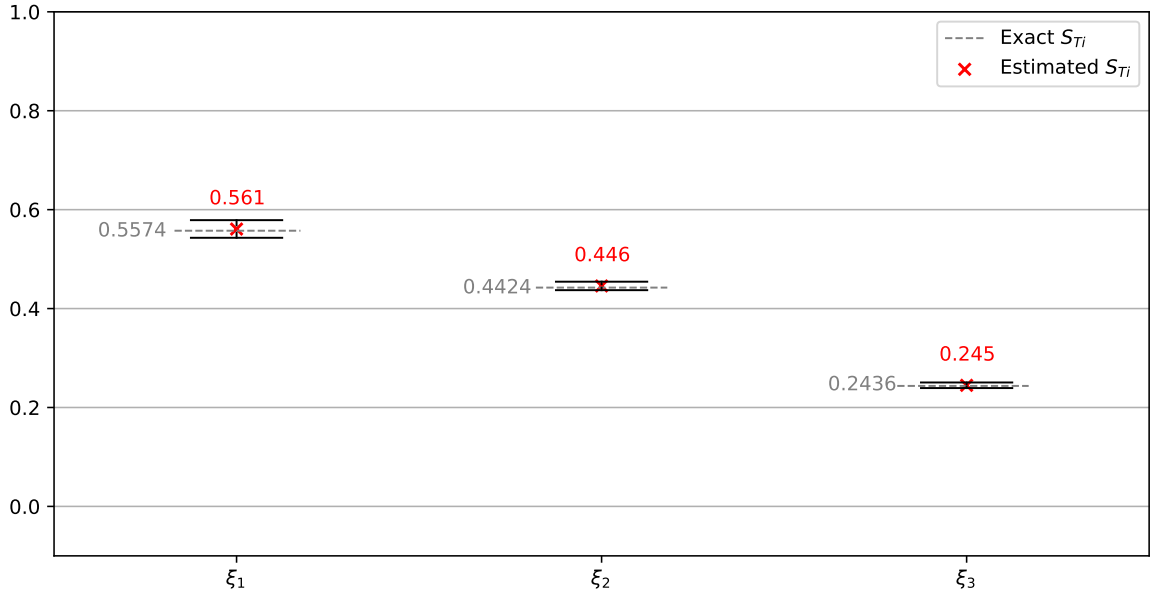


Figure 27: Total effect sensitivities S_{Ti} for the Ishigami function with 95% confidence intervals. We used a sample size of 50000 with the MC method.

A.3. Sensitivity Analysis Results (High-Dimensional)

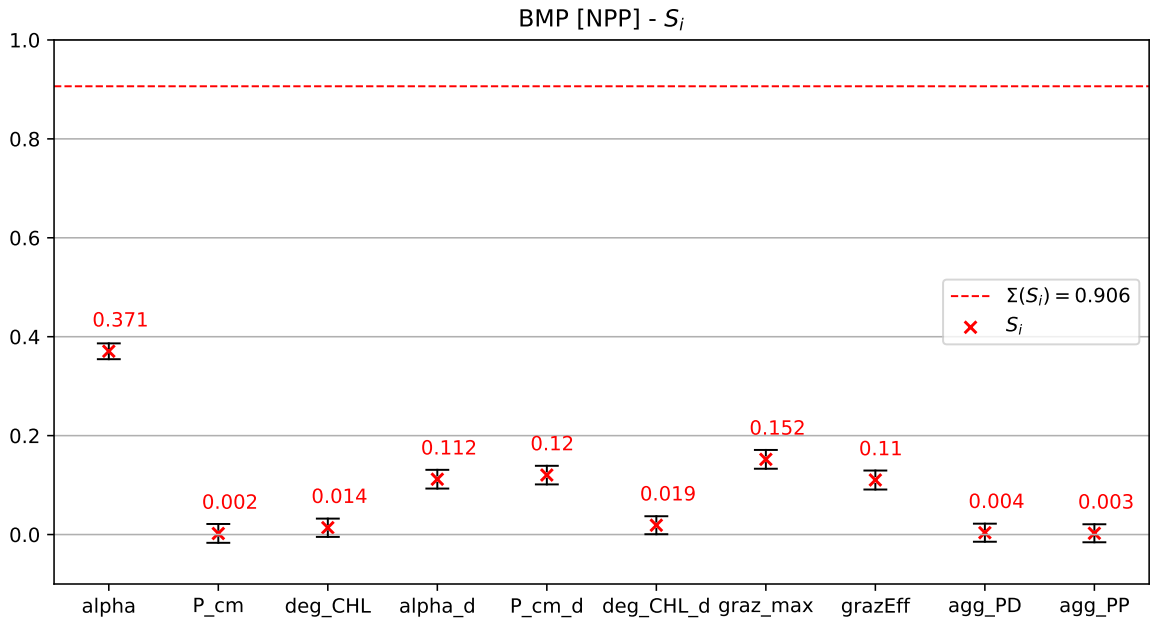


Figure 28: First order sensitivities S_i for the BMP [NPP] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

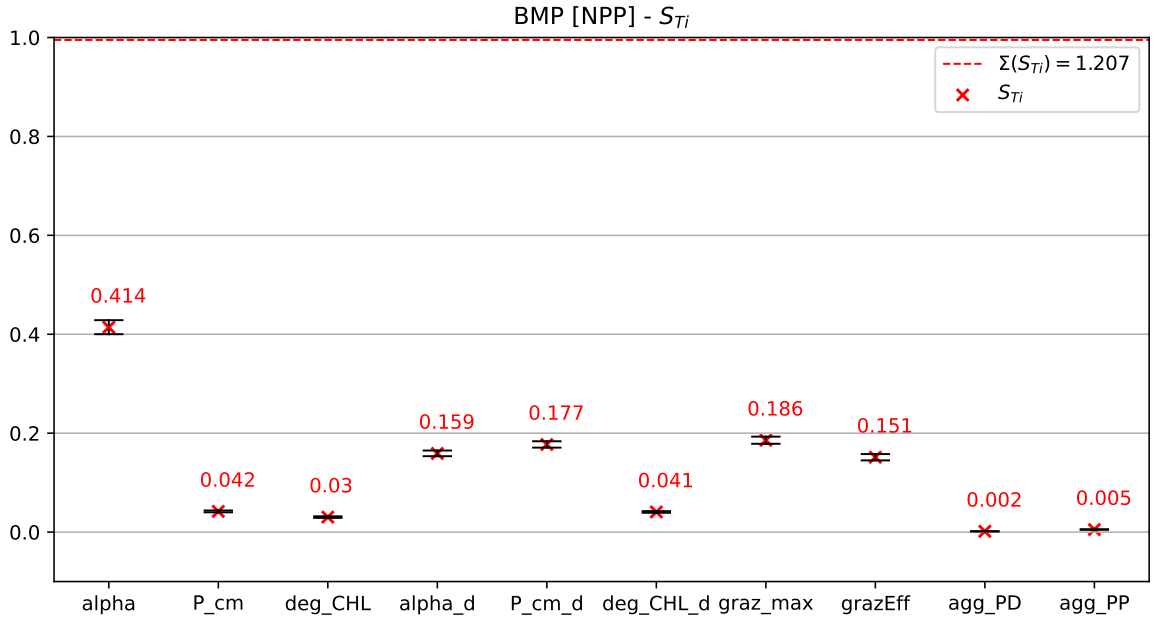


Figure 29: Total effect sensitivities S_{Ti} for the BMP [NPP] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

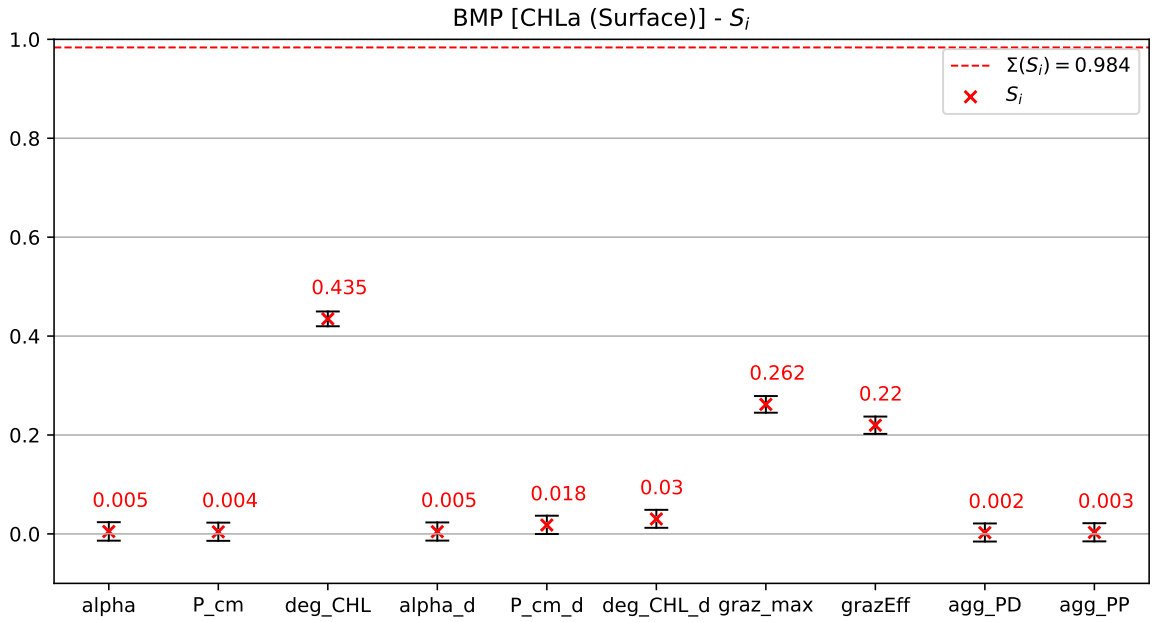


Figure 30: First order sensitivities S_i for the BMP [CHLa (Surface)] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

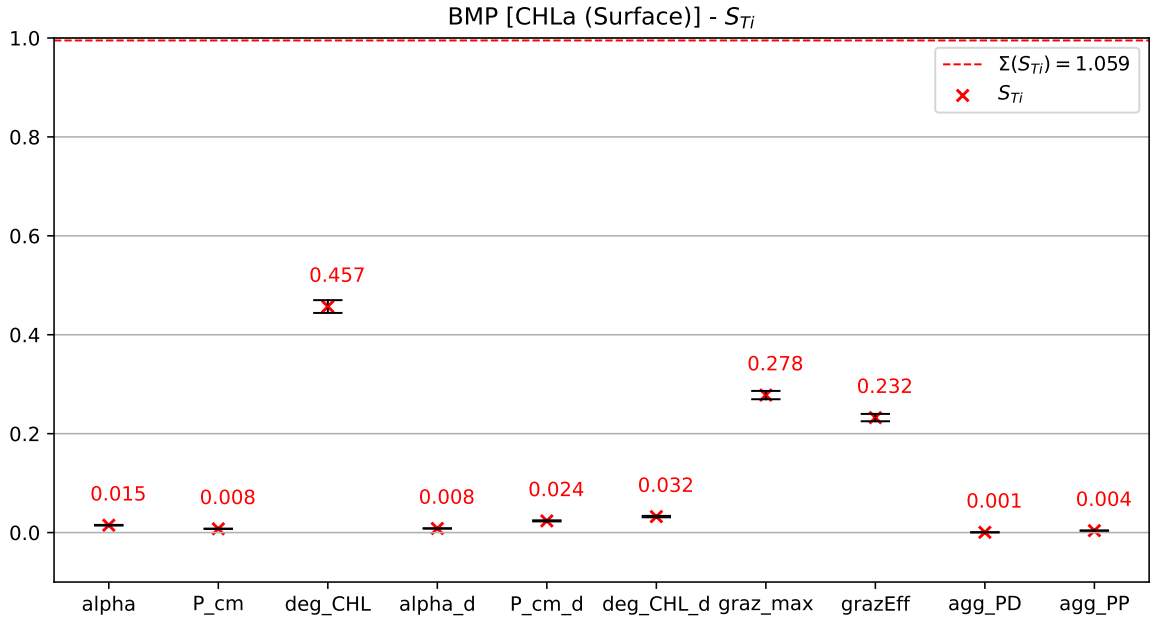


Figure 31: Total effect sensitivities S_{Ti} for the BMP [CHLa (Surface)] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

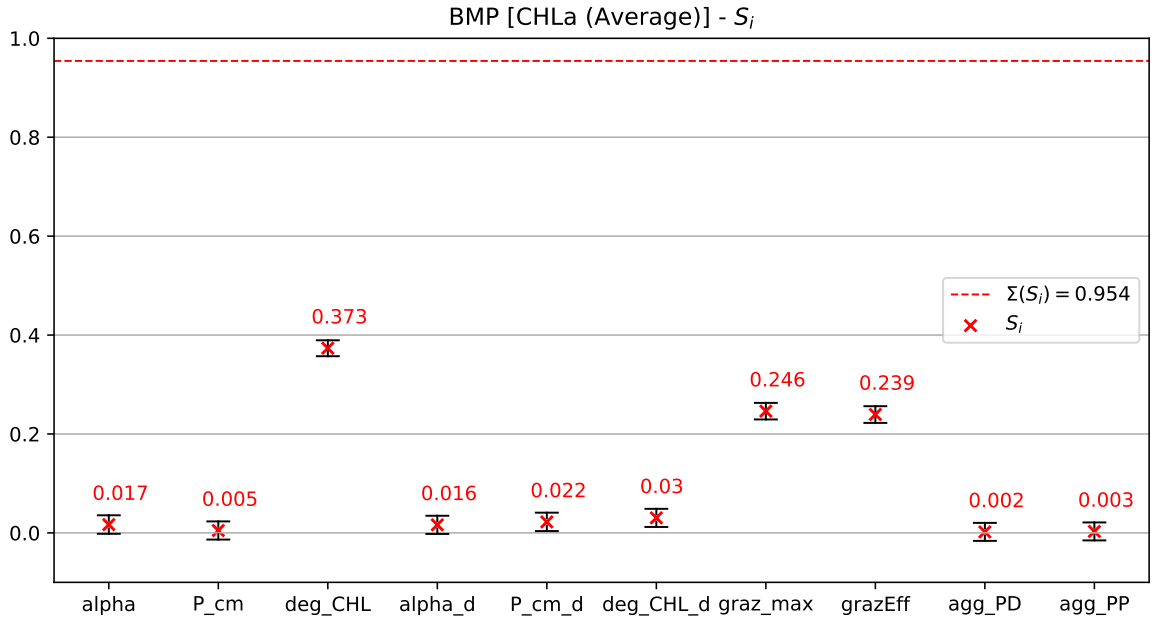


Figure 32: First order sensitivities S_i for the BMP [CHLa (Average)] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

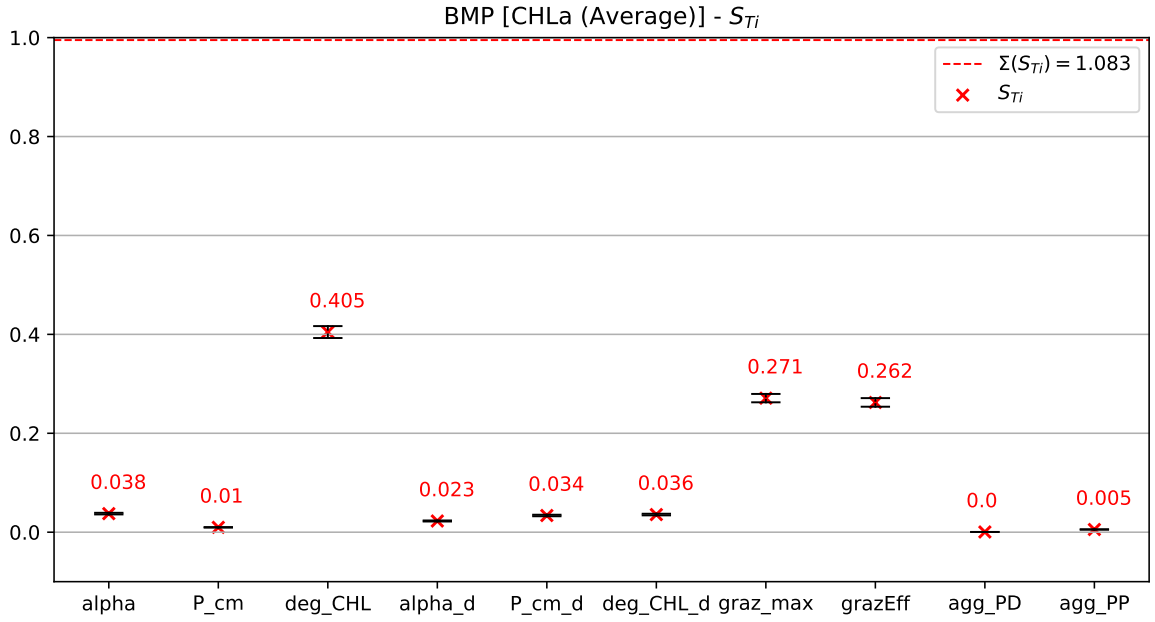


Figure 33: Total effect sensitivities S_{Ti} for the BMP [CHLa (Average)] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

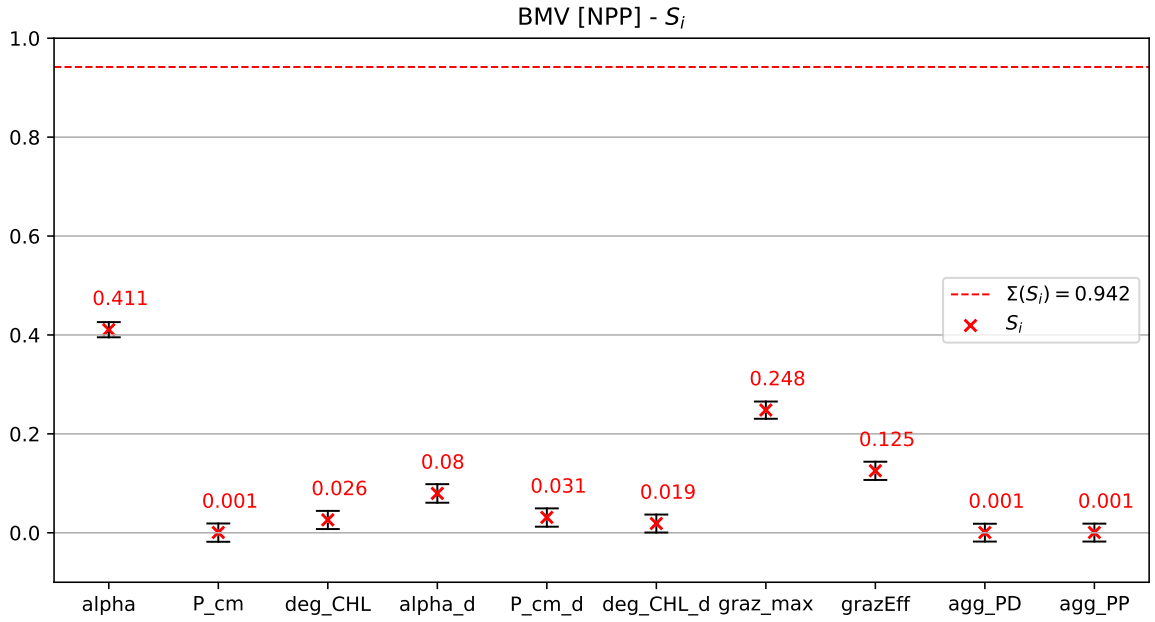


Figure 34: First order sensitivities S_i for the BMV [NPP] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

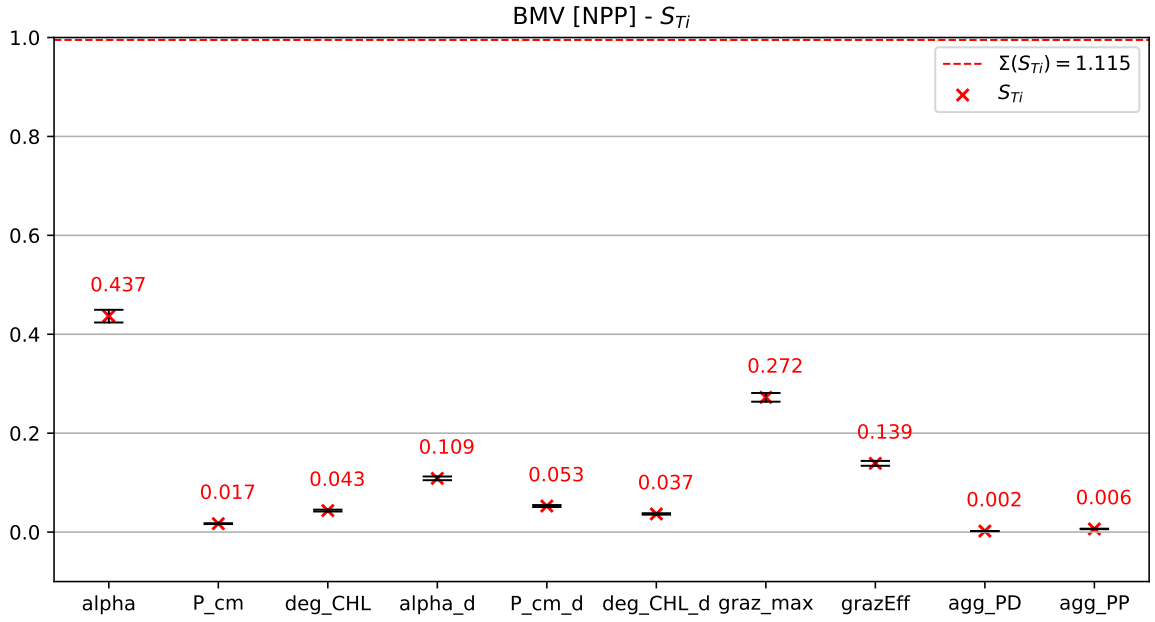


Figure 35: Total effect sensitivities S_{Ti} for the BMV [NPP] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

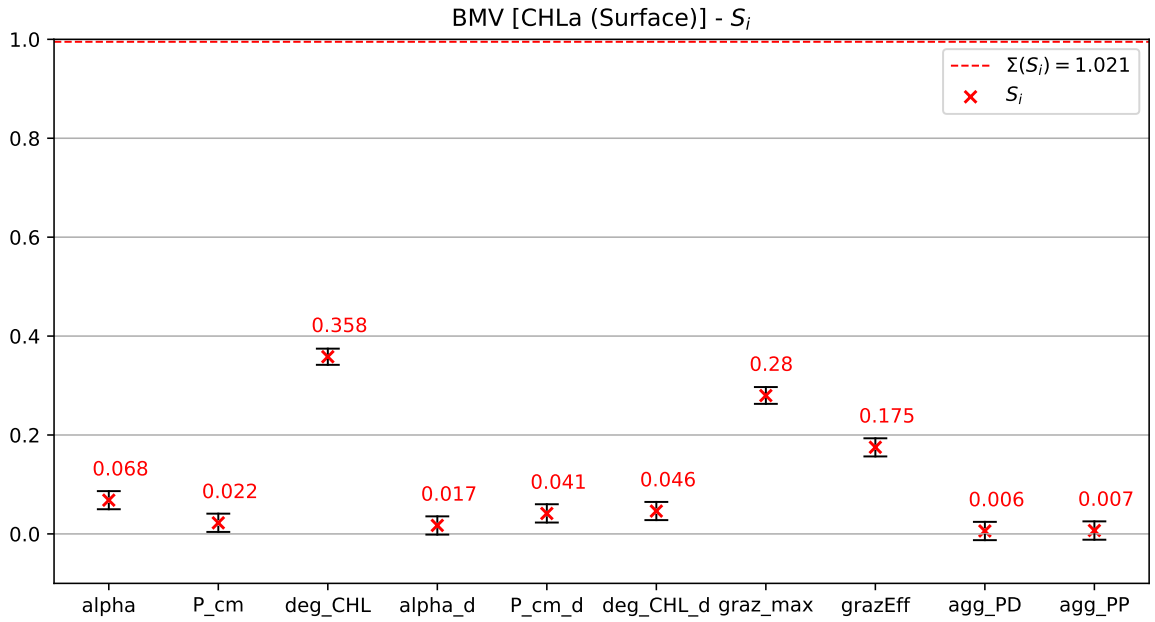


Figure 36: First order sensitivities S_i for the BMV [CHLa (Surface)] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

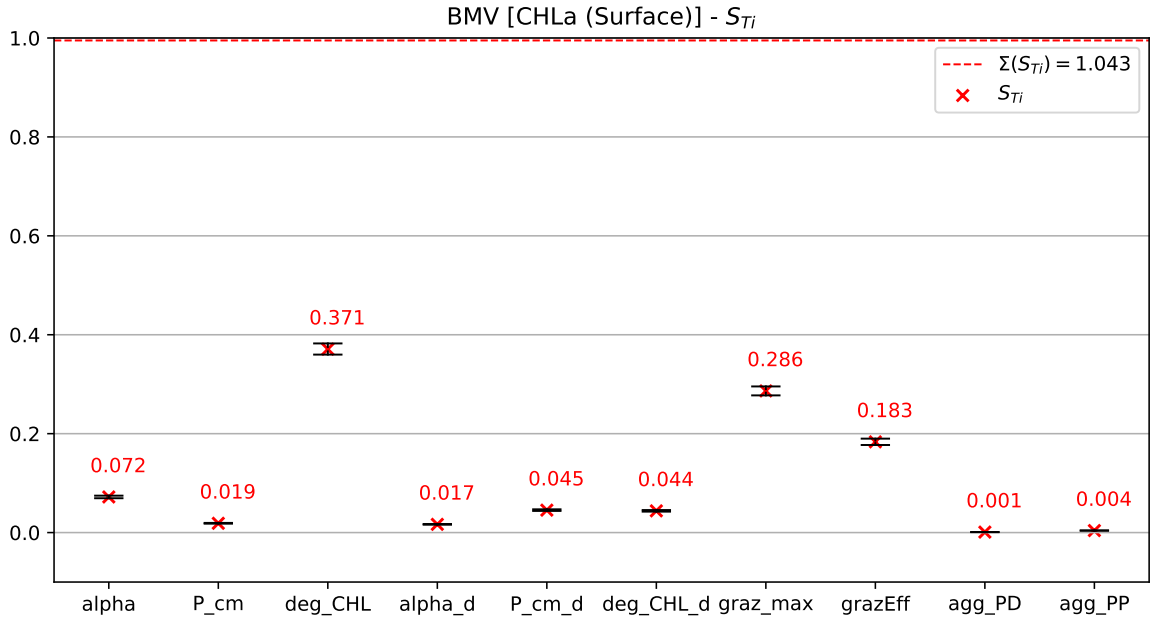


Figure 37: Total effect sensitivities S_{Ti} for the BMV [CHLa (Surface)] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

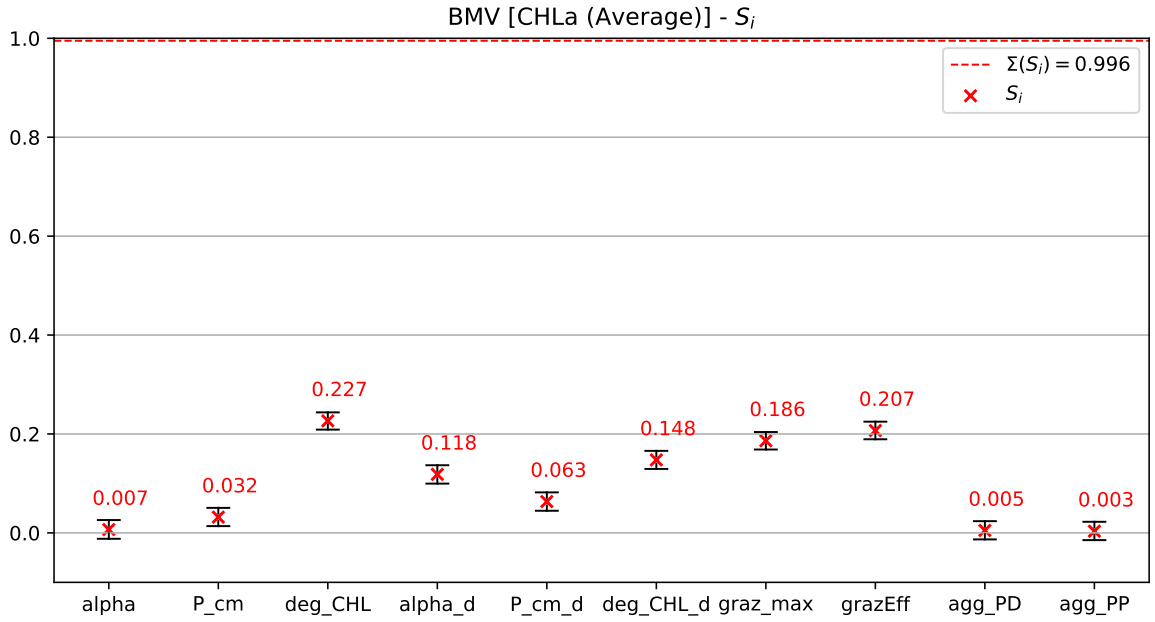


Figure 38: First order sensitivities S_i for the BMV [CHLa (Average)] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

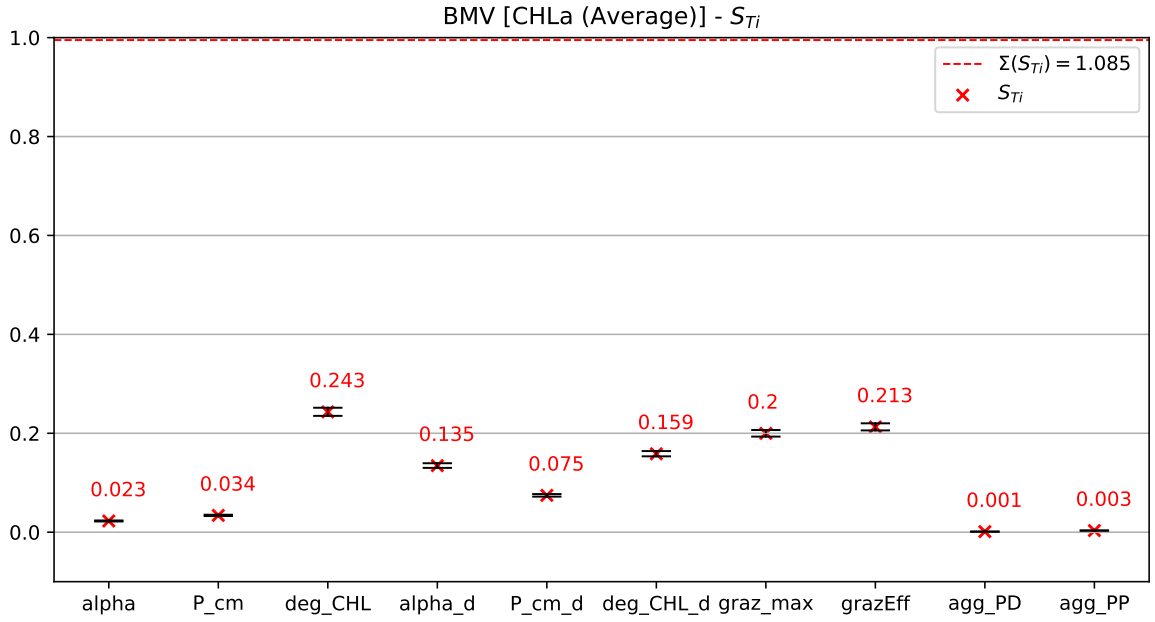


Figure 39: Total effect sensitivities S_{Ti} for the BMV [CHLa (Average)] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

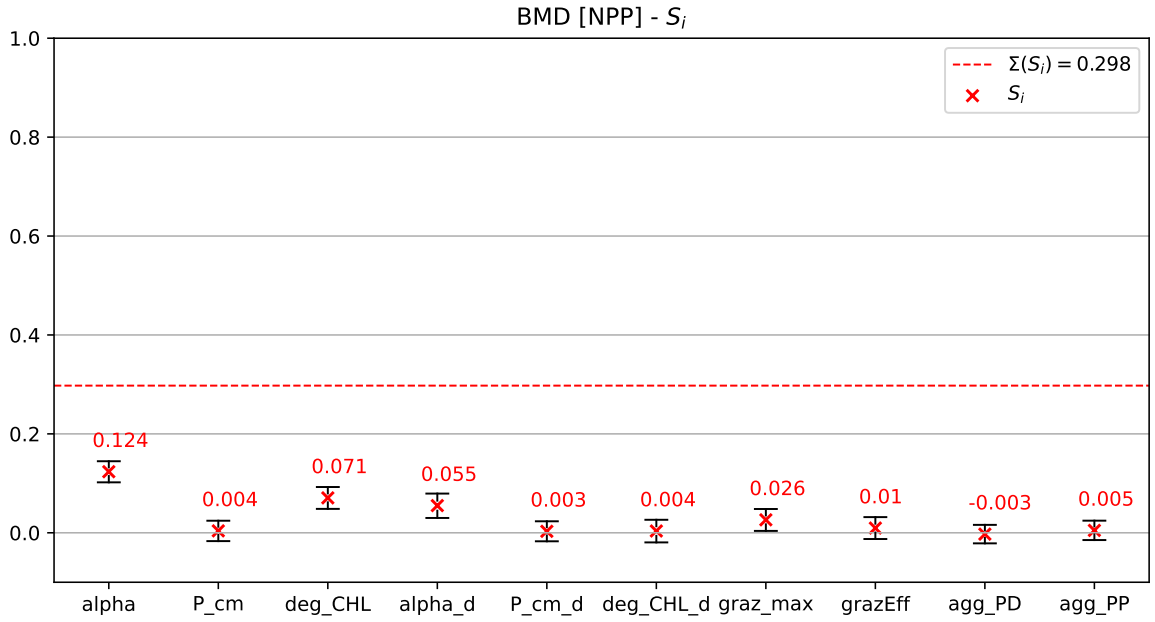


Figure 40: First order sensitivities S_i for the BMD [NPP] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

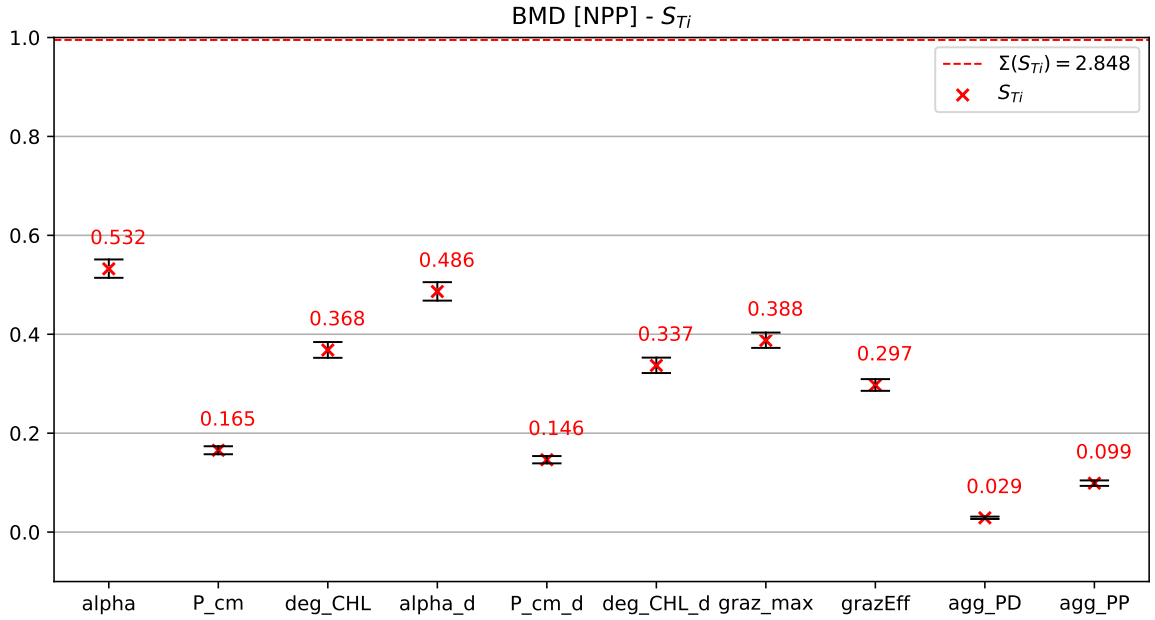


Figure 41: Total effect sensitivities S_{Ti} for the BMD [NPP] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

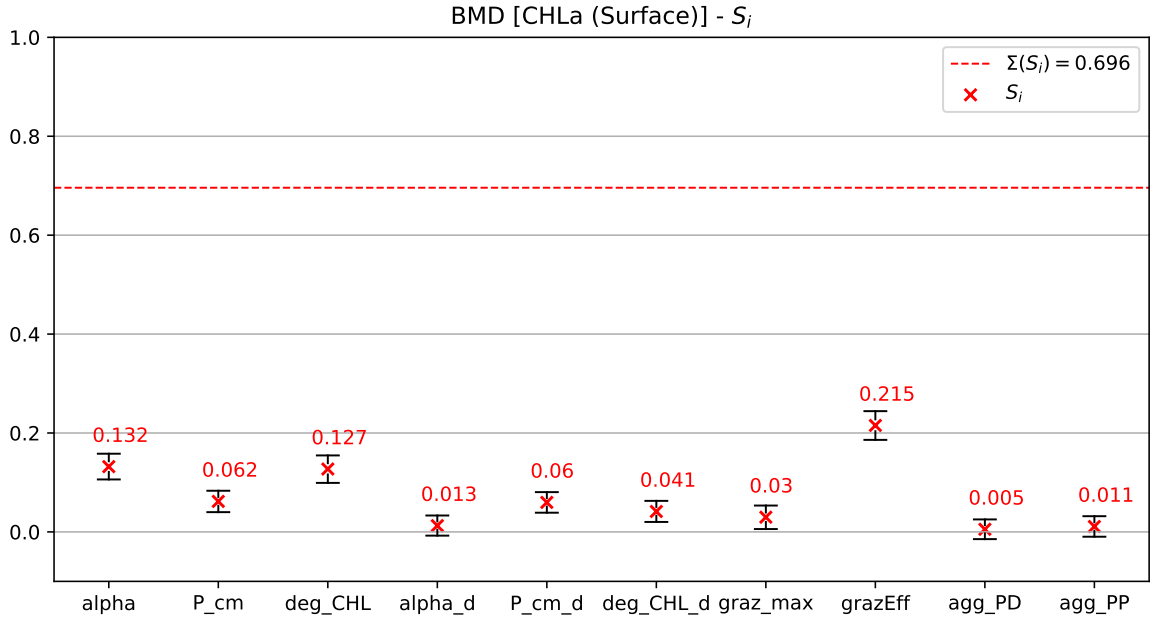


Figure 42: First order sensitivities S_i for the BMD [CHLa (Surface)] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

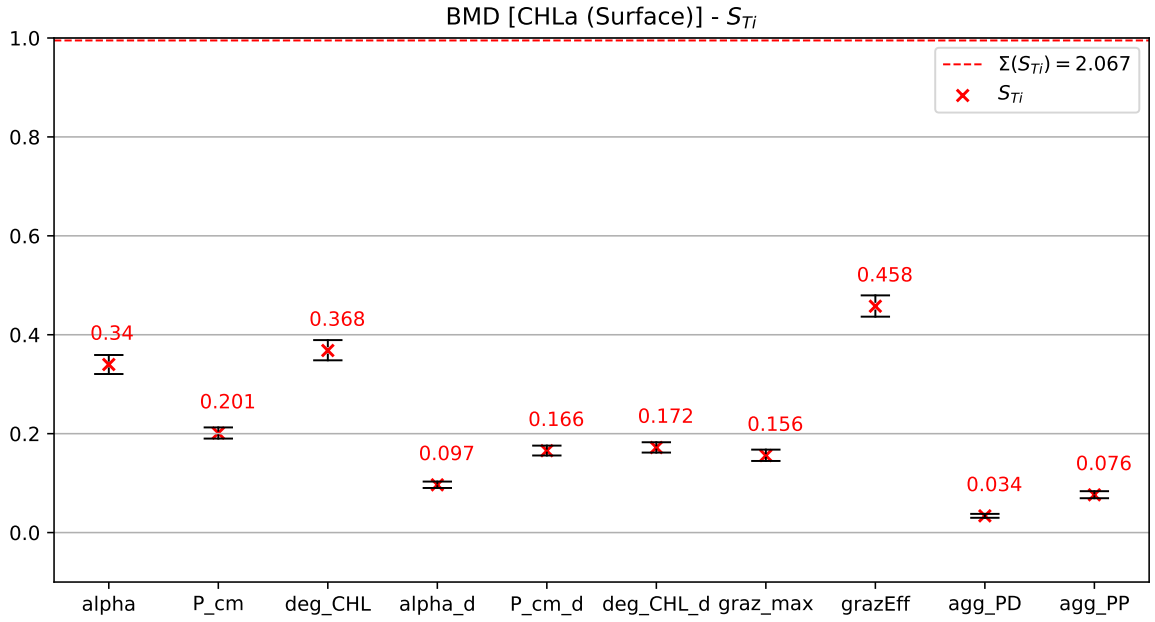


Figure 43: Total effect sensitivities S_{Ti} for the BMD [CHLa (Surface)] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

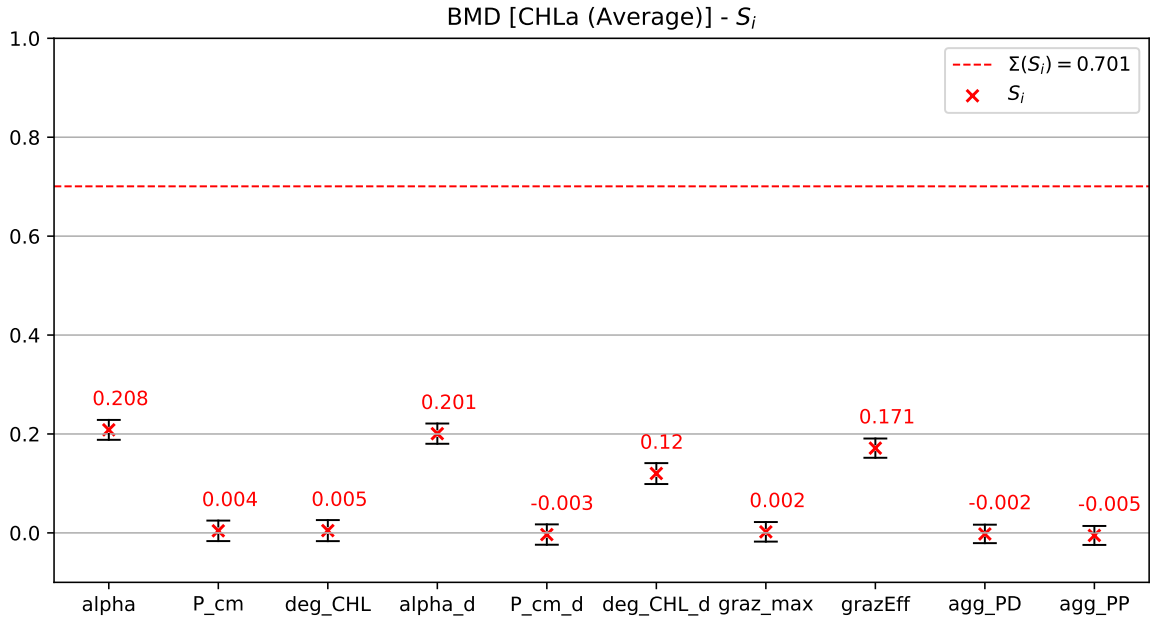


Figure 44: First order sensitivities S_i for the BMD [CHLa (Average)] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

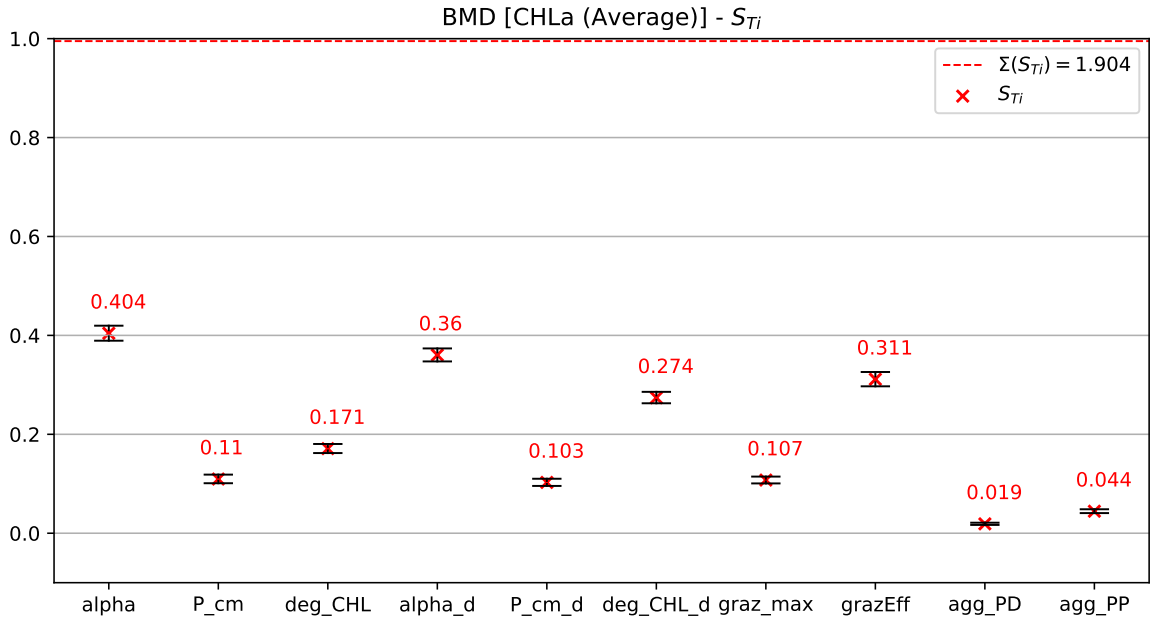


Figure 45: Total effect sensitivities S_{Ti} for the BMD [CHLa (Average)] quantity with 95% confidence intervals. We used a sample size of 25000 with the MC method.

A.4. Sensitivity Analysis Results (Low-Dimensional)

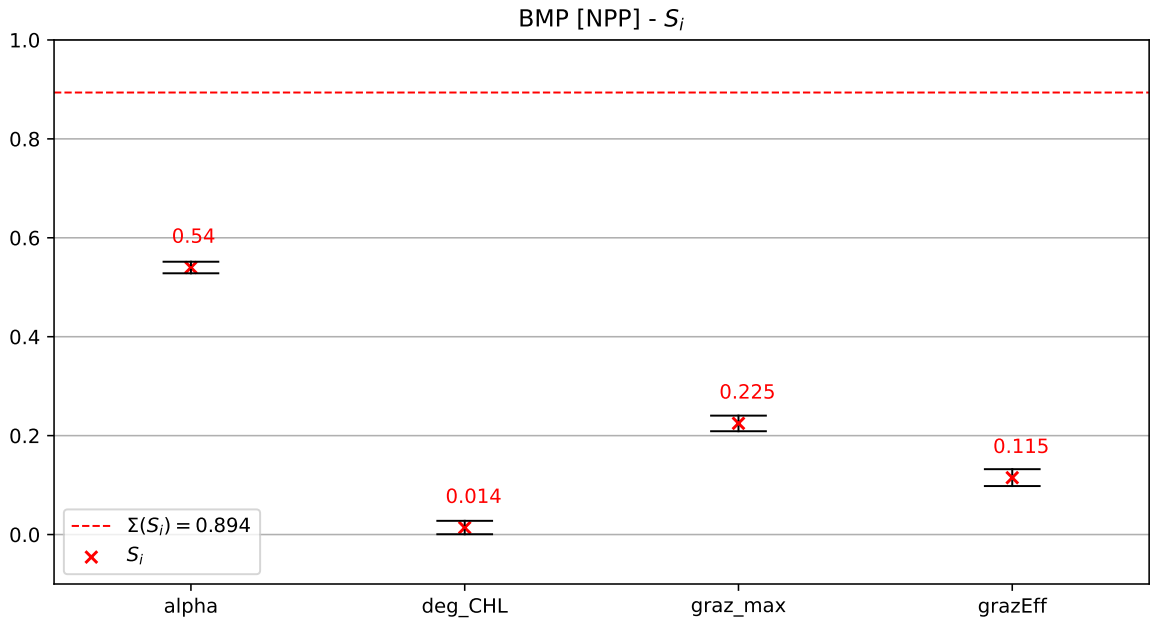


Figure 46: First order sensitivities S_i for the BMP [NPP] quantity with 95% confidence intervals. We used a sample size of 50000 with the MC method.

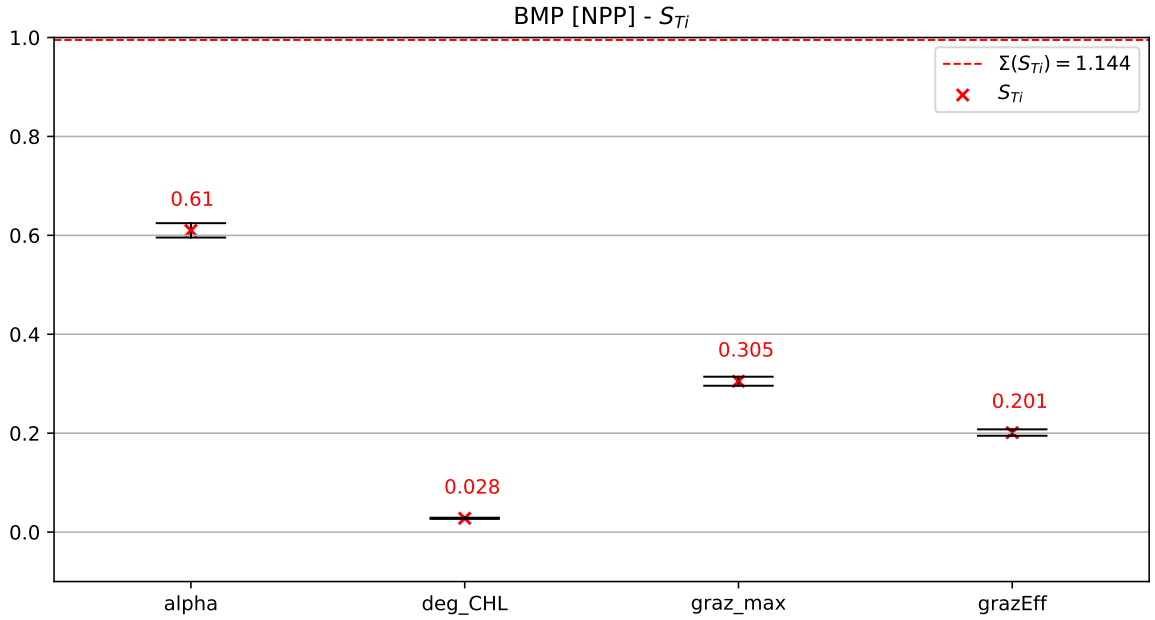


Figure 47: Total effect sensitivities S_{Ti} for the BMP [NPP] quantity with 95% confidence intervals. We used a sample size of 50000 with the MC method.

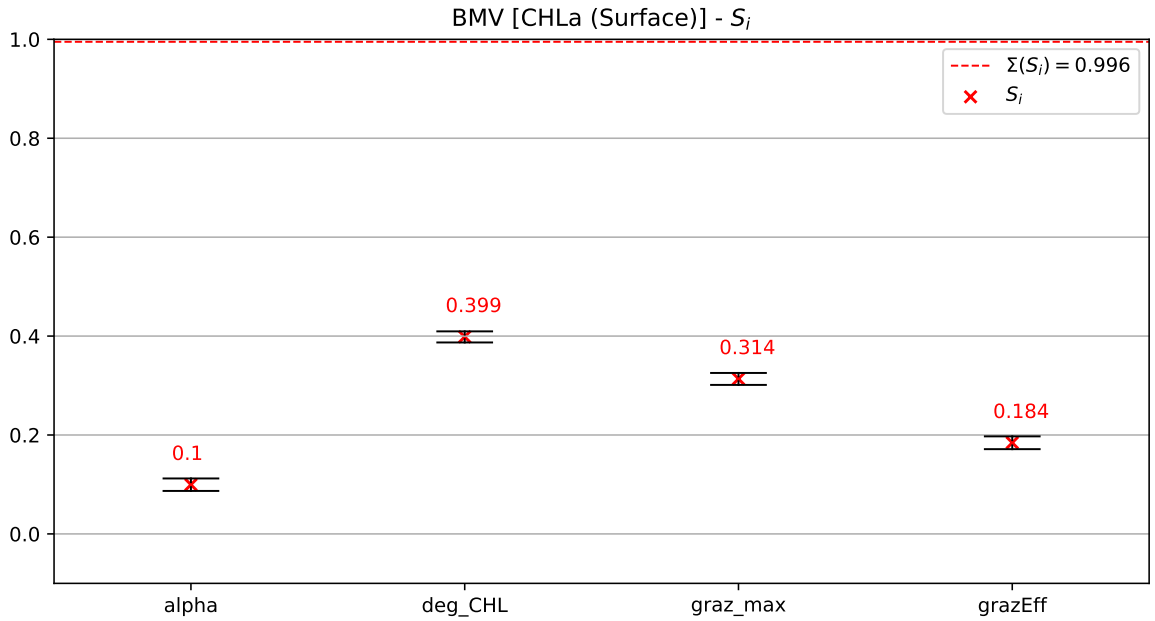


Figure 48: First order sensitivities S_i for the BMV [CHLa (Surface)] quantity with 95% confidence intervals. We used a sample size of 50000 with the MC method.

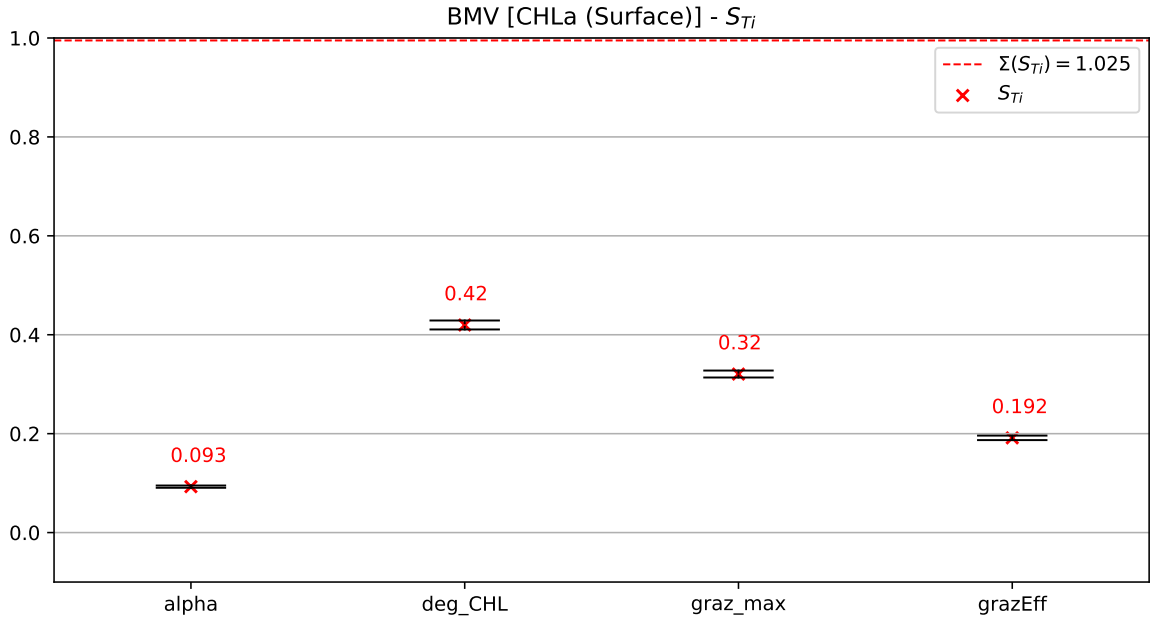


Figure 49: Total effect sensitivities S_{Ti} for the BMV [CHLa (Surface)] quantity with 95% confidence intervals. We used a sample size of 50000 with the MC method.

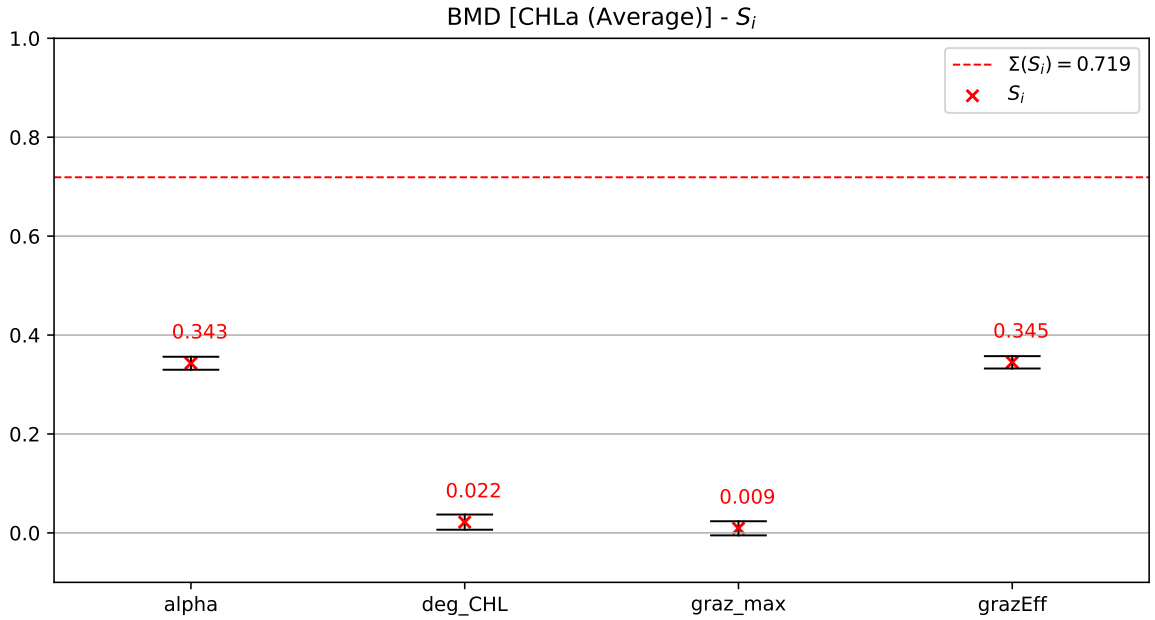


Figure 50: First order sensitivities S_i for the BMD [CHLa (Average)] quantity with 95% confidence intervals. We used a sample size of 50000 with the MC method.

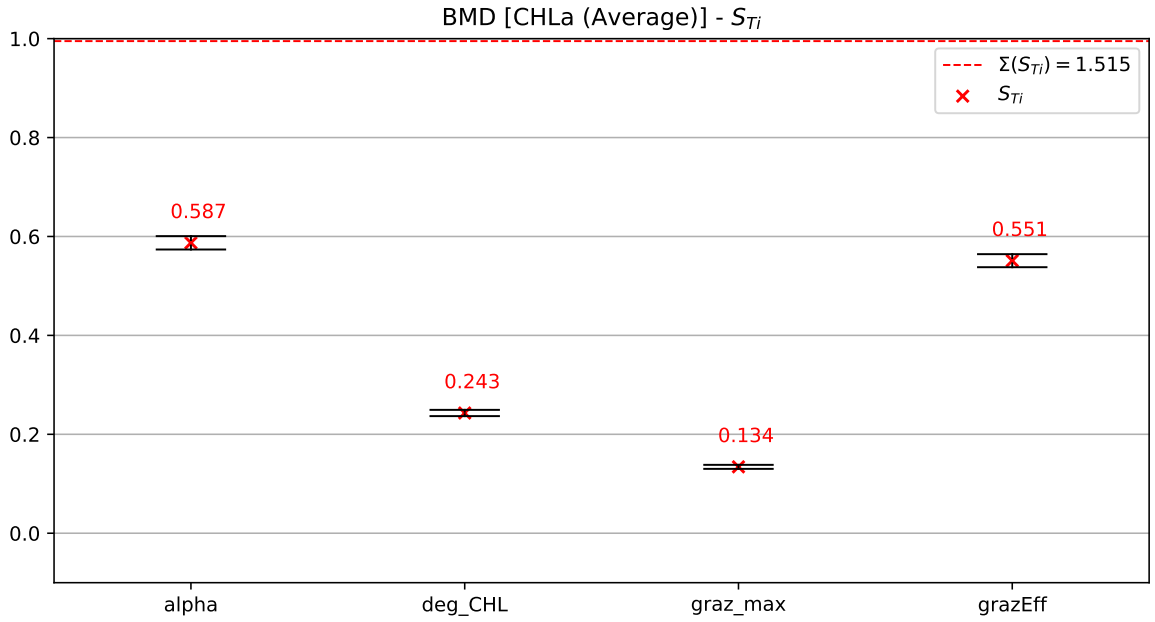


Figure 51: Total effect sensitivities S_{Ti} for the BMD [CHLa (Average)] quantity with 95% confidence intervals. We used a sample size of 50000 with the MC method.

A.5. Sensitivity Analysis Convergence

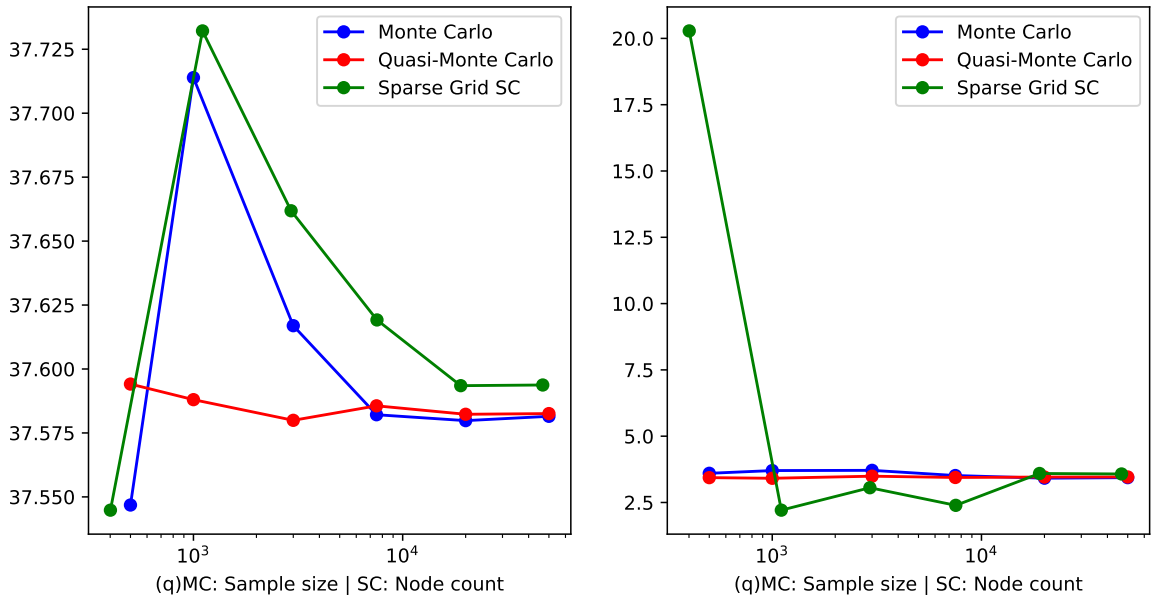


Figure 52: Convergence rates of expectation value (left) and variance (right) for the BMP [NPP] quantity.

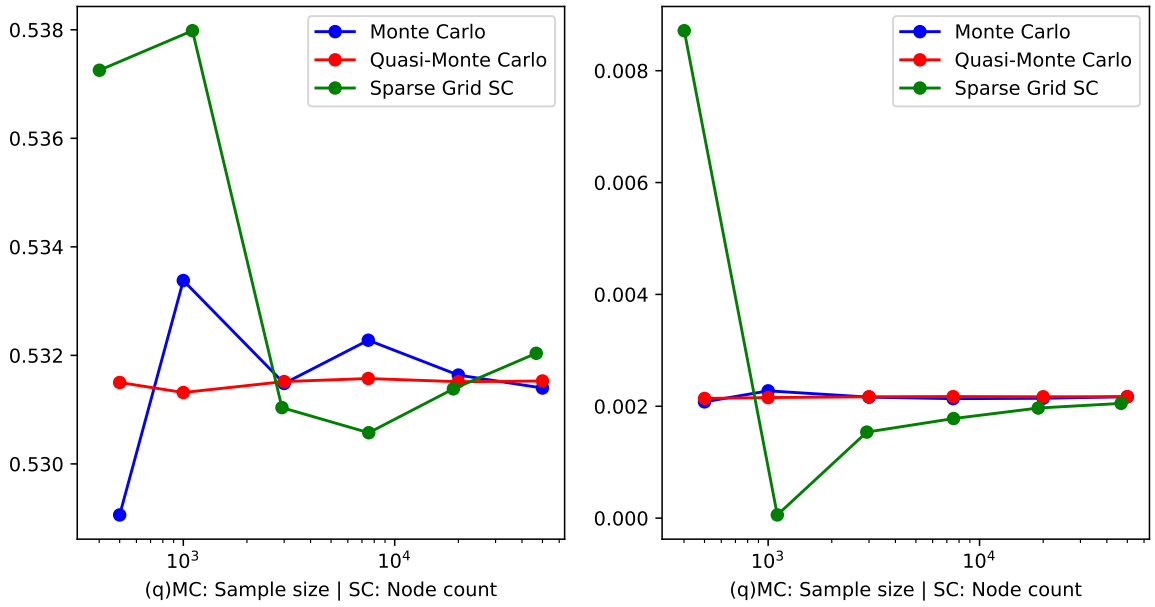


Figure 53: Convergence rates of expectation value (left) and variance (right) for the BMP [CHLa (Surface)] quantity.

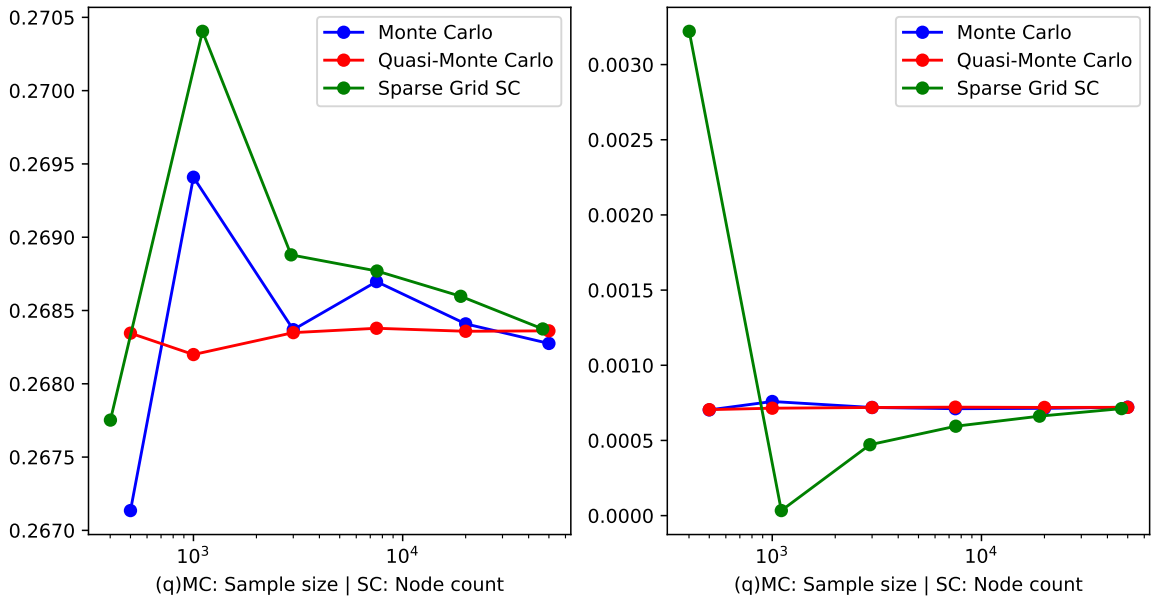


Figure 54: Convergence rates of expectation value (left) and variance (right) for the BMP [CHLa (Average)] quantity.

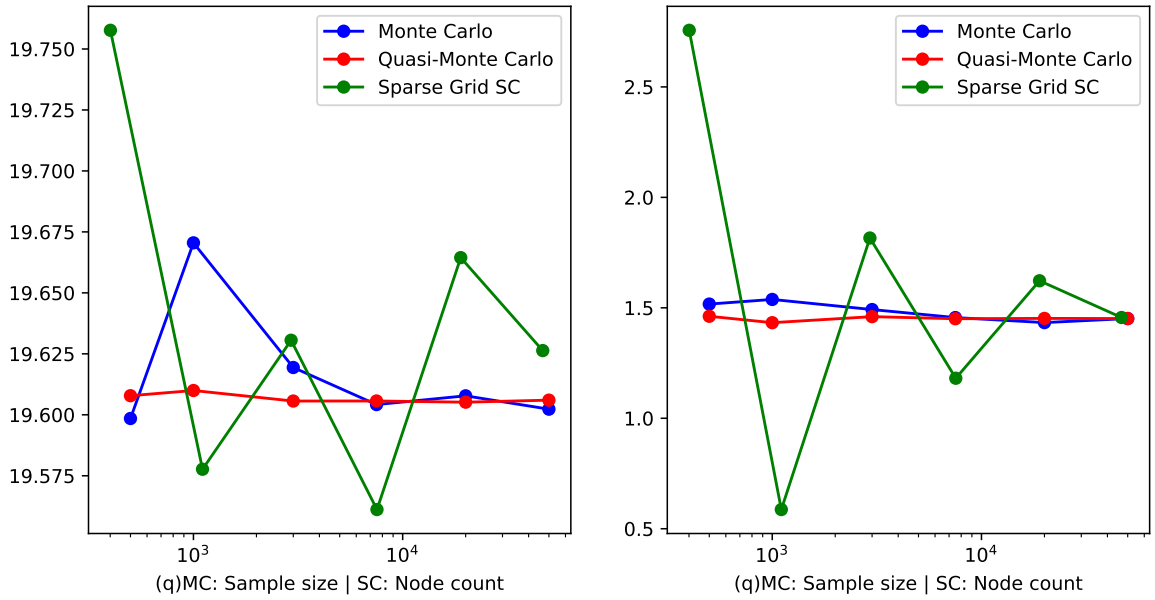


Figure 55: Convergence rates of expectation value (left) and variance (right) for the BMV [NPP] quantity.

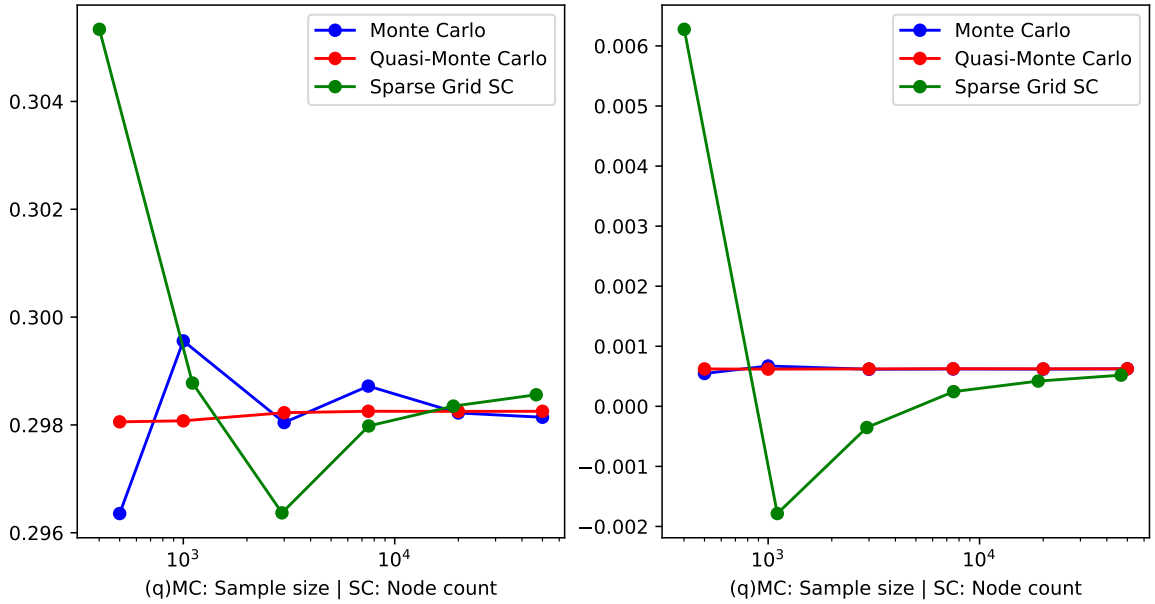


Figure 56: Convergence rates of expectation value (left) and variance (right) for the BMV [CHLa (Surface)] quantity.

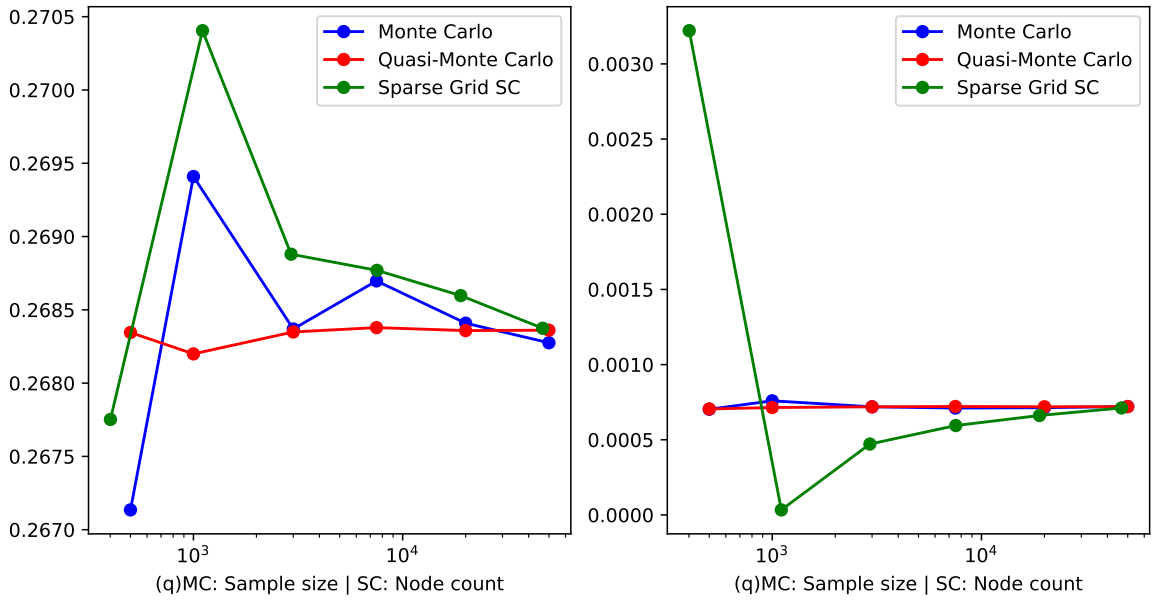


Figure 57: Convergence rates of expectation value (left) and variance (right) for the BMV [CHLa (Average)] quantity.

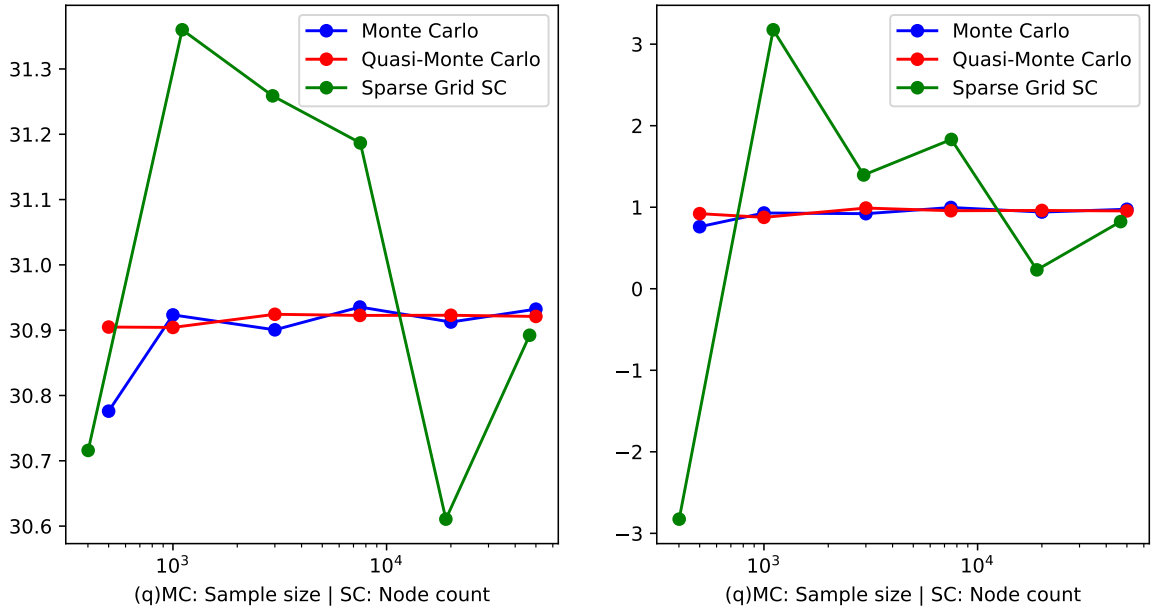


Figure 58: Convergence rates of expectation value (left) and variance (right) for the BMD [NPP] quantity.

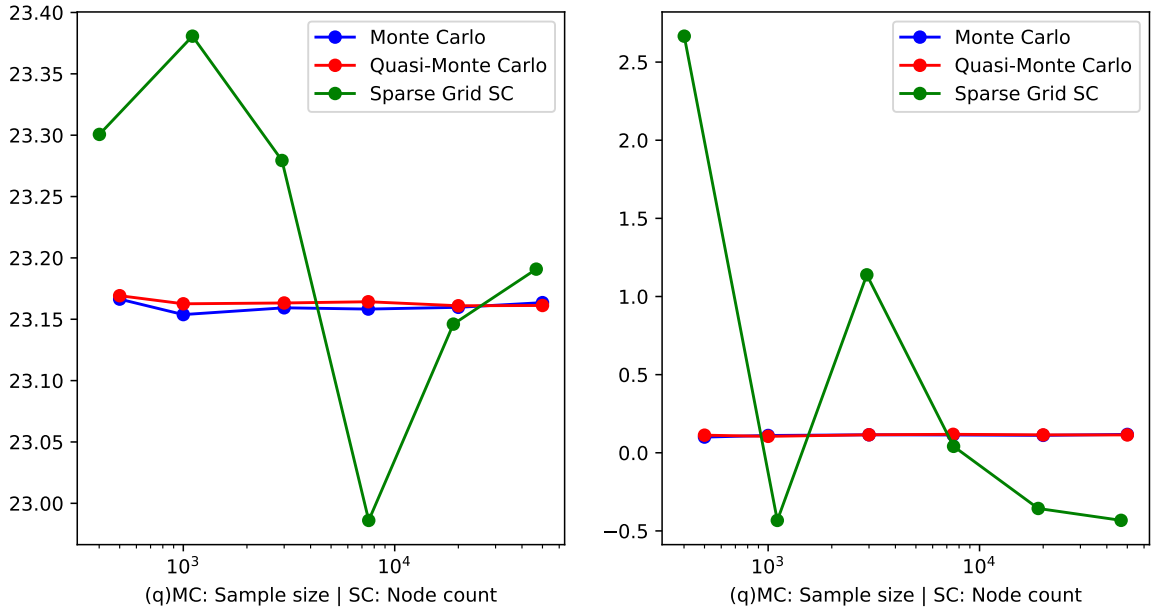


Figure 59: Convergence rates of expectation value (left) and variance (right) for the BMD [CHLa (Surface)] quantity.

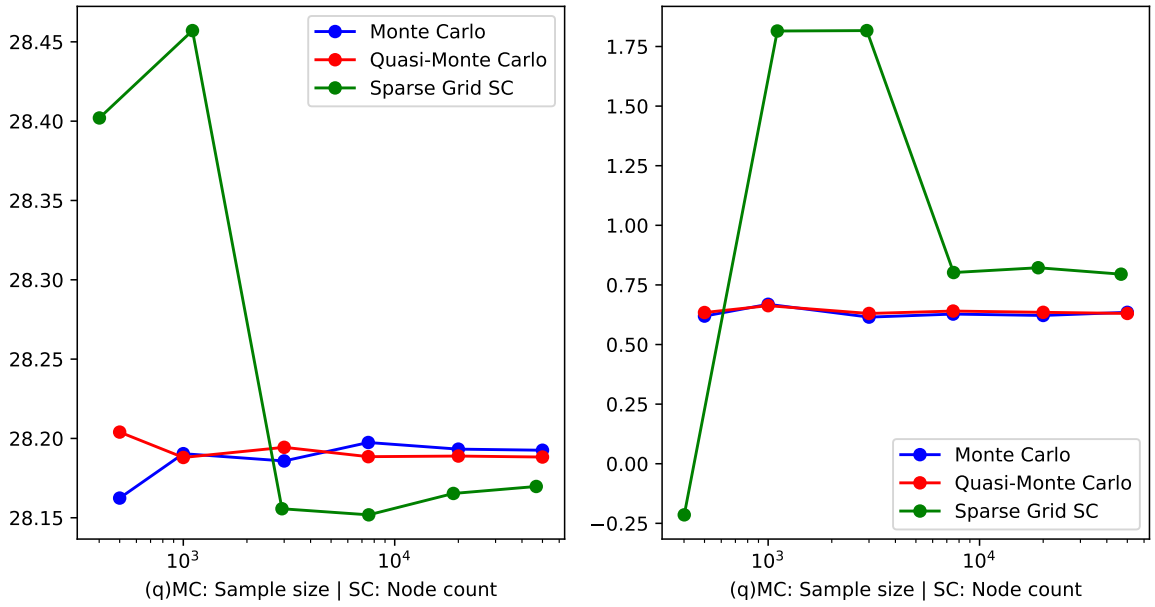


Figure 60: Convergence rates of expectation value (left) and variance (right) for the BMD [CHLa (Average)] quantity.