Analyzing and parameterizing the PRELES model

Simon Schulte (Mat.nr: 3320165)
Umweltsystemmodellierung SS2016
Dozent: Dr. Florian Hartig
17. Juli 2016

Contents

1 Introduction

Ecosystem models are widely used to predict photosynthetic production under climate change (Potter et al. 1993; Sitch et al. 2003; Wolfgang Cramer et al. 1999; Melillo et al. 1993). Here, we use the PRELES model to predict gross photosynthetic production (GPP) under a worst-case climate change scenario. For model calibration we take a two-year time sequence of daily observed GPP values from a site in Finland . As model input we use daily weather data over the same period of time.

PRELES is a equation-based forest model of intermediate complexity (Peltoniemi, Pulkkinen, et al. 2015). The model is semi-empiric and has been developed and parametrized under boreal conditions (Peltoniemi, Kalliokoski, Lindroos, et al. 2012). PRELES predicts daily gross primary production (GPP), evapotranspiration (ET) and soil water content (SW). It requires input data in form of daily mean temperature (T), vapor pressure deficit (VPD), precipitation, photosynthetic photon flux density (PAR), and fraction of absorbed PAR (fAPAR).

The model has been calibrated to data from different sites in Finland. It has been used for predictions on regional scale where it showed good performance (Minunno et al. 2014). Regarding prediction of GPP the model delivered a similar outcome compared to the JSBACH model - another forest model (Peltoniemi, Markkanen, et al. 2014). PRELES was also used as a sub-model to investigate the consequences of adaptive resource allocation of trees under climate change conditions (Peltoniemi, Kalliokoski, and Mäkelä 2015).

Model description Figure 1 shows the system's state variables, the main processes and the model inputs. A description of the parameters can be found in table 3 of the annex. The model consists of two sub-models: the photosynthesis of the canopy (in form of GDD) and the water balance of the ecosystem. Both sub-systems are interlinked. The GDD model describes the accumulation of atmospheric carbon by photosynthesis. This process is influenced by the input variables VPD, temperature, PAR, fAPAR, along with other subsidiary processes and parameters. The influence of temperature is via a first-order dynamic delay model in order to delay the model's response on current temperature, as well as to capture seasonality.

The water balance sub-model consists of three water stocks: canopy water, soil water and snow water. The canopy water stock gets filled by precipitation and becomes soil water when its maximum amount is reached. Snow water content increases by precipitation if $T < 0°C$ and melts if $T > 0°C$. Soil water drains if field capacity is reached. The storages of all three water stocks are sequentially decreased by evapotranspiration: First canopy water, then snow water and finally soil water. The evapotranspiration process is furthermore influenced by VPD, PAR, soil water and other subsidiary processes and parameters.

Between the sub-models there exist interlinkages in both direction: First, there is the influence of soil water content on photosynthesis of the canopy. This process leads via the so-called stomatal conductance, where basically two

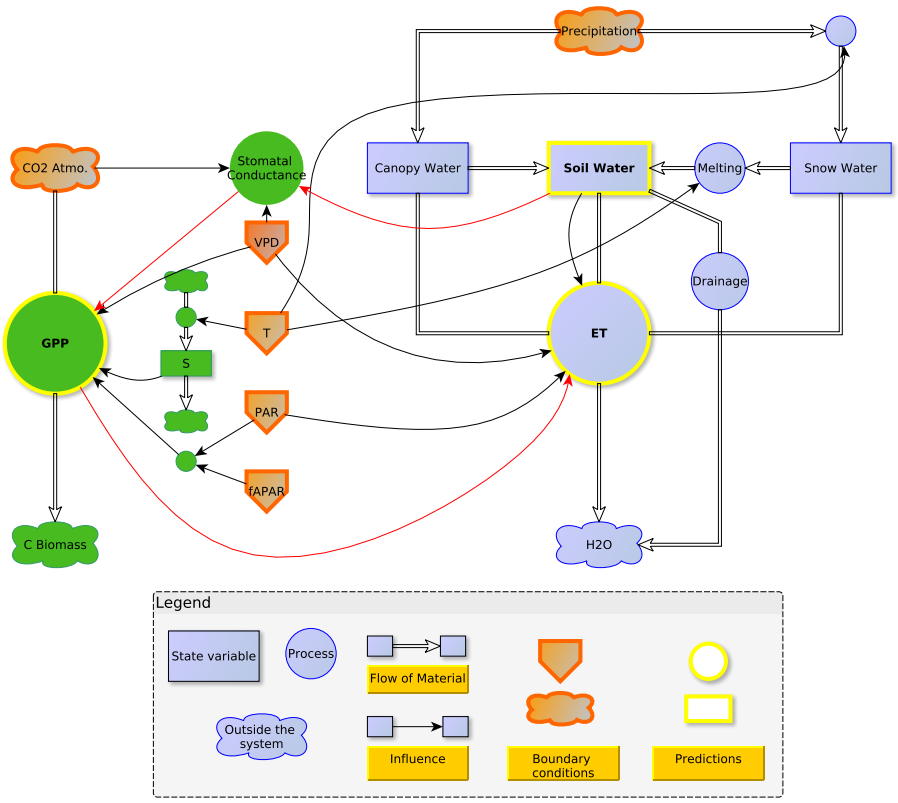


Figure 1: Simplified diagram of the PRELES model structure. GPP = Gross Primary Production, ET = Evapotranspiration, T = Temperature, VPD = Vapour Pressure Deficit, PPFD = Photosynthetic Photon Flux Density, S = State of acclimation.

processes compete: On the one hand, the amount of extractable water which is directly positively linked to soil water content. On the other hand a process which is negatively influenced by VPD and positivley affected by CO_2-concentration. Thus, a high amount of soil water only increases GPP up to a certain point, after which this effect is constraint by a high VPD and/or a low CO_2-concentration (and the other way round).

Secondly, there is an positive influence of photosynthesis in the canopy on evapotranspiration. That means, increasing GPP leads to an increased evapotranspiration. There also exist indirect linkages through the input variables VPD, temperature and PAR which affect both sub-models.

2 Sensitivity analysis (SA)

In order to determine the effect small perturbations of the parameters and the boundary conditions (in the following referred to as factors) have on the model output a sensitivity analysis (SA) is conducted.

a) Local SA

For a first overview we carry out a local sensitivity analysis. As there are some fixed parameters we focus only on those with a range of possible values (see annex, table 2).

In the first step we have to define standard values for both, parameters and boundary conditions. For the parameters we take the given reference values (see annex, table 2), while in case of the boundary conditions we take the mean values of the observed time sequence. Based on these values one factor at a time is changed $+10\%$ and -10%. Using this new parameter set we predict GPP and compare it to the reference value derived from predicting with standard combination of parameter and boundary conditions.

In the results we see that only three parameters cause a change in the prediction of GPP (see figure 2): beta (potential light use efficiency) has a positive linear effect on GPP. Conversely the gamma-parameter (light modifier for saturation with radiance) has a negative effect on photosynthetic production. A small negative change in the output is caused by the kappa-parameter (regulating the strength of the influence of VPD on GDD). For all other parameters local SA shows no effect on the model outcome. In case of the boundary conditions we observe positive effects of PAR and fAPAR on GPP and a marginal negative effect of VPD.

However, local sensitivity analysis is limited concerning factor interactions and nonlinearity of factor effects. Thus, we conduct a global sensitivity analysis in form of a morris screening.

b) Global SA - Morris screening

The morris method randomly walks in the entire parameter space changing one factor at a time. The range in which the factors are varied is determined by their defined minimum/maximum values in case of the parameters (see annex, table 2), and by the minimum/maximum values within the time sequence in case of the boundary conditions. For each step the changes in GPP prediction are saved. The mean and the standard deviation of the resulting distributions give us a measure of the sensitivity of each factor regarding the model output. Here, we take 1000 iterations, 10 levels in which the factors vary (within their specific range) and a grid.jump of 5.

The results show that gamma is the most sensitive parameter followed by beta, X[0], kappa, S[max] and tau (see Figure 2, left). Changes in all other parameters have no effect on GPP prediction. In case of the boundary conditions

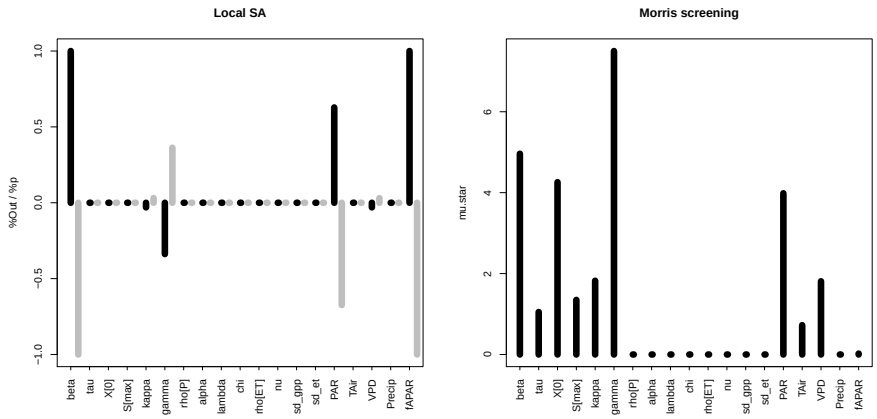


Figure 2: Left: Results of the local sensitivity analysis on parameter and boundary conditions. Grey bars represent -10% from reference value, Black bars +10%. Right: Results of Morris screen after 1000 iterations. It shows the mean of the change of GPP when changing the parameter.

PAR is most sensitive followed by VPD and TAir. Changing precipitation or fAPAR has no effect on GPP prediction.

Comparing the results of local and global sensitivity analysis shows two differences: First, there are some factors which show no sensitivity in the local SA, while in global SA they do. This can be explained by nonlinearity and interactions of these factors: As the local SA tests at only one value (the reference value), it does not show if a factor is sensitivity in a different part of its range (nonlinearity). Nor it shows up when the factor is only sensitive with a special combination of one (or more) other parameters (interaction). Second, there is one boundary condition (fAPAR, fraction of absorbed PAR), which is sensitive in the local SA, while in the global SA it is not. This is due to the small range of this factor in our data, which is exceeded by the 10% changes in the local SA.

Those parameters causing no change in GPP prediction when being varied are only part of the water balance sub-model.

3 Parameter calibration and validation

a) Calibration

In order to calibrate the parameters to the data, we use the Bayesian Markov-Chain-Monte-Carlo method (MCMC). For sampling the differential evolution algorithm is taken (DEzs in r package BayesianTools). Based on a prior distribution and a likelihood function MCMC determines a posterior distributions for each parameter.

Here, we only take the parameters having shown sensitivity in the morris

Table 1: Reference value, minimum, maximum and calibrated values of selected parameter.

	reference	min	max	calibrated (mean)
gamma	0.03	0.00	0.50	0.06
S[max]	18.77	2.30	30.00	19.04
tau	13.23	1.00	25.00	7.11
X[0]	-3.97	-20.00	20.10	-3.05
beta	0.75	0.20	2.50	0.96
c/sd_gpp	1.00	0.00	5.00	0.14
a	1.00	0.00	5.00	0.20
b	1.00	0.00	5.00	0.01

screen: beta, X[0], tau, S[max] and gamma, as well as the three parameters a,b, sd_gpp of the likelihood function (see table 1). In a first step, we use a normal distribution $\mathcal{N}(\mu, \sigma^2)$ within the likelihood function with constant standard deviation σ. As prior distribution we choose uniform distributions for each parameter as there is no more information available except minimum and maximum values. We make 200,000 iterations with three Markov-Chains. Convergence of the chains is checked both, visually and by the Gelman and Ruben diagnostics (Brooks and Gelman 1998).

b) Validation

In a first step our assumptions for the likelihood function have to be validated. Therefore, we compare the model predictions with the observed data. First, we have a look at the Bayesian p-values over time. There, a clear pattern can be observed (see figure 4, top-left): During the winter days the p-values cluster around 0.5, while in summer the values spread between 0 and 1 with light clusters at 1. This observation is confirmed by a histogram of the p-values (see figure 4, top-right).

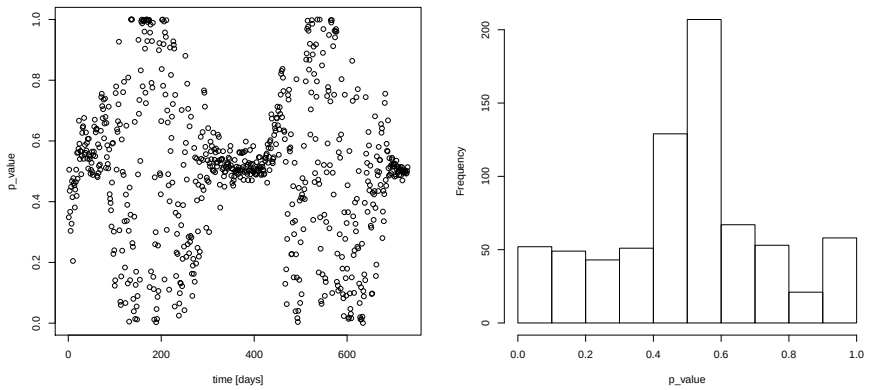
Consequentially, we have to change our likelihood function in a way that our predictions fits better the low data variance in winter and the high variance in summer. As the variance of the data seems to be influenced by GPP (see figure 5) we include a quadratic dependence of GPP on the σ value of the normal distribution: $\mathcal{N}(\mu, \sigma^2)$ with $\sigma = a \cdot GPP^2 + b \cdot GPP + c$.

Now, the temporal pattern in the p-values is less distinct and the histogram is flatter (figure 4, bottom). Thus, we favor this error distribution and have a closer look at its p-values and possible autocorrelation.

In figure 4 there can still be observed a light pattern of p-values regarding posterior prediction of GPP. For low GPP predictions the p-values cluster between 0.5 and 1, which confirms the previously discussed light temporal pattern (figure 4, bottom).

Temporal autocorrelation of the p-values can be observed, too (see figure 4, top-right). This means that temporal close GPP values are not independent.

constant σ



quadratic dependence of GPP on σ

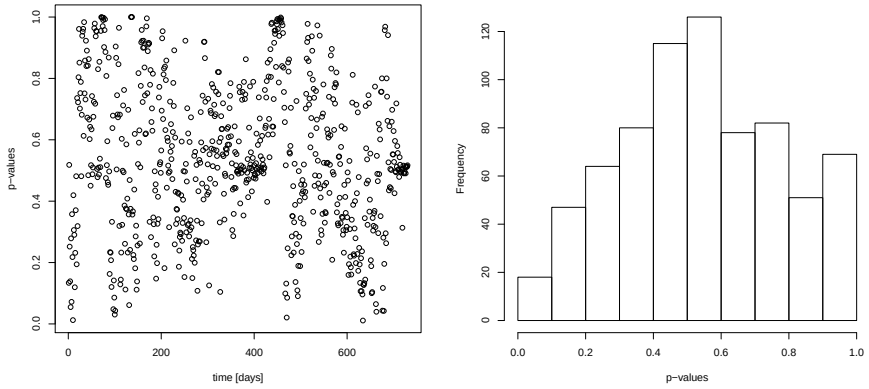


Figure 3: Bayesian p-value plotted over time with 1st of January as day 1 (left) and the histogram of the p-value (right). Figures on top are with normal error distribution with constant σ, figures below with quadratic dependence of σ from predicted GPP.

However, this is rather unsurprising as the model inputs are climate variables, that are normally not independent neither.

There can be also observed a light systematic model bias: For high observed GPP values the model underestimates photosynthetic production (see figure 4, bottom-left). When plotting predicted GPP against observed GPP, we see that all in all the model fits well and explains 96% of the deviance (figure 4, bottom-right). However, the systematic model bias can also be observed here.

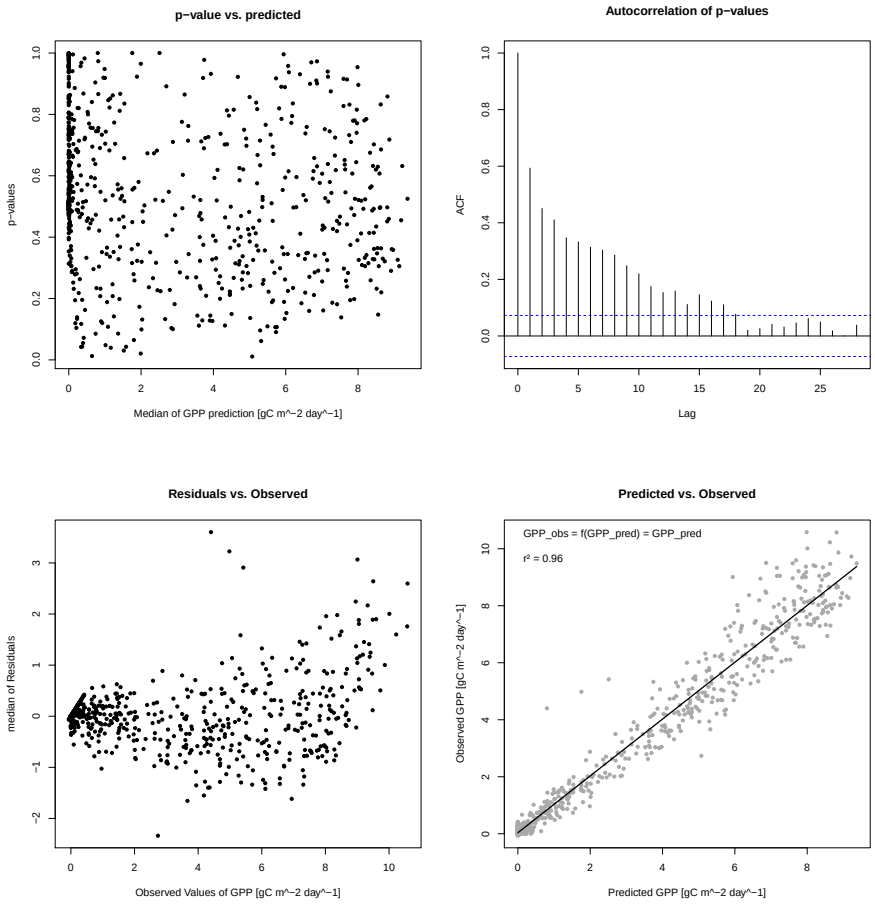


Figure 4: Top-left: p-values plotted against the median of predicted GPP in order to determine heteroscedasticity. Top-right: Autocorrelation of p-values. Bottom-left: Observed GPP plotted against residuals to visualize model bias. Bottom-right: Predicted GPP plotted against observed GPP.

4 Prediction uncertainty

In the next step, the posterior distributions of the parameters are analyzed and then used in order predict on the observed data, as well as on a climate change scenario.

In figure 5 the uncertainty of the parameters can be observed. In case of beta and S[max] the uncertainty is relatively high. However, this uncertainty can be explained by the strong correlation of these parameters ($r = 0.96$, see figure 5, left). The dependence of beta and S[max] seems to be due their interactions: High potential light use efficiency requires a high threshold value for maximum

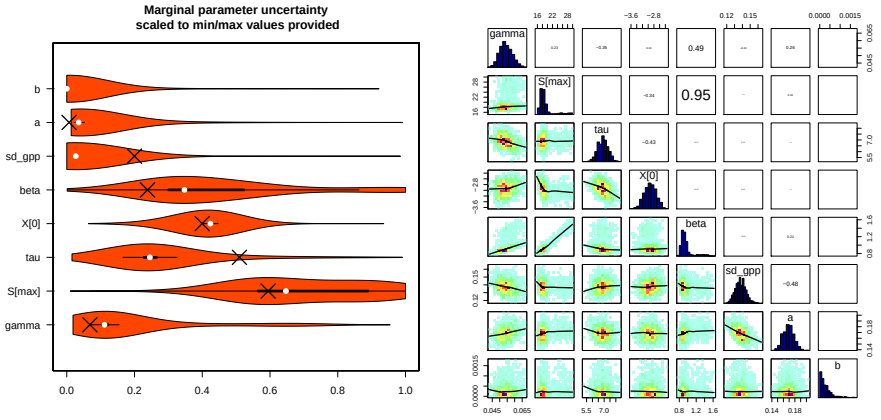


Figure 5: Left: Marginal parameter uncertainty scaled to min/max values after MCMC with 200,000 iterations. Right: Correlation matrix of parameters.

acclimation.

Prediction on data The posterior distributions can now be taken to predict to our input data. We draw 1834 parameter combinations out of the posterior distribution and run the model with each. In order to get a 95 % confidence interval we compute the 2.5 % and the 97.5 % quantiles for each time step. The uncertainty of our predictions results from the previously defined error function. The result can be observed in figure 6. There the effect of the quadratic dependence of the σ in our error function can be seen: Prediction uncertainty in summer is much higher than in winter which fits well the data.

Prediction on climate change Using the same parameter combinations but varying the input data we can now predict on different climate conditions. We take the worst-case climate projections for Finland from Jylhä, Tuomenvirta, and Ruosteenoja (2004) for the year 2050 with the baseline period of 1961–1990. They project an increase in annual mean temperature by 5 °C and an increase in annual mean precipitation by 30 %. The projected atmospheric CO_2 concentration for this year is derived from IPCC Assessement Report 4 where they project a mean CO_2 concentration of 512 ppm (Solomon 2007).

With this scenario the model predicts a mean increase in GPP of 41 % (figure 6, right). Regarding winter (from 11/07 to 05/07) GPP more than doubles on average (+151 %). In summer (from 05/08 to 11/06) the increase is 30 %. However, uncertainty on the prediction is high.

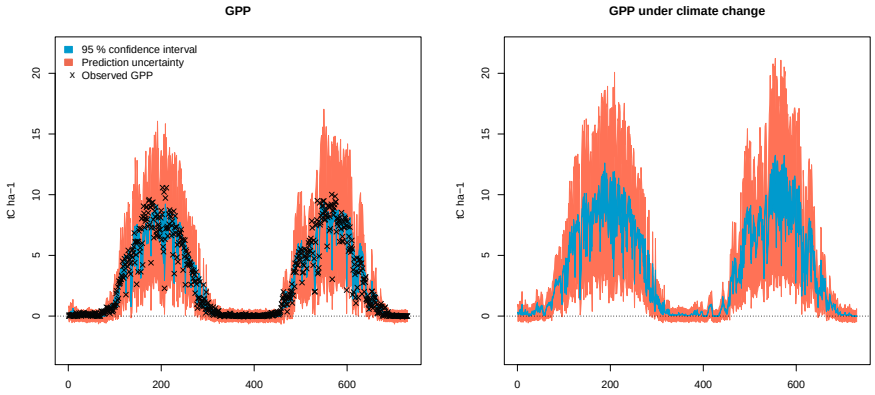


Figure 6: Left: Predictions on the data. Right: Predictions on climate change (+5°C temperature, +30% precipitation, CO_2 concentration of 512 ppm), x-axis: Days with 1st of January as day 1.

5 Conclusion

With a normal error distribution and a quadratic dependence of σ of GPP we get a good fit to the data. In winter. however, prediction uncertainty partly still ranges in areas below zero. This could be fixed by taking a strictly positive error distribution - for example the lognormal distribution.

The uncertainty in our predictions only covers uncertainties in the parameters and the initial conditions. However, neither uncertainties in boundary conditions (e.g. measuring uncertainty), nor structural uncertainties in the model processes are included.

In case of the uncertainty of the prediction under climate change there must be considered that PRELES is a semi-empiric model. Therefore, it may perform well on regional level and under the current boundary conditions. However, as there is no observed data for climate change, processes and/or parameters in the models may change under different conditions. For instance, trees growing in an CO_2 enriched air show a reduced sensitivity to VPD and droughts (Heath 1998). Another uncertainty not covered in the prediction, is the interactions of climate change with the boundary conditions VPD, PAR and fAPAR, which here have been assumed to be constant.

Lastly, the problem with spatial autocorrelation could be encountered by taking a autoregressive model (ARn) within the likelihood function.

References

Brooks, Stephen P and Andrew Gelman (1998). "General methods for monitoring convergence of iterative simulations". In: *Journal of computational and graphical statistics* 7.4, pp. 434–455.

Cramer, Wolfgang et al. (1999). "Comparing global models of terrestrial net primary productivity (NPP): overview and key results". In: *Global change biology* 5 (S1), pp. 1–15.

Heath, J (1998). "Stomata of trees growing in CO2-enriched air show reduced sensitivity to vapour pressure deficit and drought". In: *Plant, Cell & Environment* 21.11, pp. 1077–1088.

Jylhä, Kirsti, Heikki Tuomenvirta, and Kimmo Ruosteenoja (2004). "Climate change projections for Finland during the 21 st century". In: *Boreal Environment Research* 9.2, pp. 127–152.

Melillo, Jerry M, A David McGuire, David W Kicklighter, Berrien Moore, Charles J Vorosmarty, and Annette L Schloss (1993). "Global climate change and terrestrial net primary production". In: *Nature* 363.6426, pp. 234–240.

Minunno, Francesco, Mikko Peltoniemi, Samuli Launiainen, and Annikki Mäkelä (2014). "Integrating ecosystems measurements from multiple eddy-covariance sites to a simple model of ecosystem process-Are there possibilities for a uniform model calibration?" In: *EGU General Assembly Conference Abstracts*. Vol. 16, p. 10706.

Peltoniemi, Mikko, Tuomo Kalliokoski, Antti-Jussi Lindroos, Egbert Beuker, and Annikki Mäkelä (2012). *User guide for PRELES, a simple model for the assessment of gross primary production and water balance of forests*. Working Papers of the Finnish Forest Research Institute.

Peltoniemi, Mikko, Tuomo Kalliokoski, and Annikki Mäkelä (2015). "Consequences of acclimating/adaptive tree CN resource balance on soil and forest carbon stocks under climate change". In: *EGU General Assembly Conference Abstracts*. Vol. 17, p. 11557.

Peltoniemi, Mikko, Tiina Markkanen, Sanna Härkönen, Petteri Muukkonen, Tea Thum, Tuula Aalto, and Annikki Mäkelä (2014). "Parallel estimates of gross primary production of Finnish forests-comparison of two process models". In: *EGU General Assembly Conference Abstracts*. Vol. 16, p. 5949.

Peltoniemi, Mikko, Minna Pulkkinen, Mika Aurela, Jukka Pumpanen, Pasi Kolari, and Annikki Mäkelä (2015). "A semi-empirical model of boreal-forest gross primary production, evapotranspiration, and soil water–calibration and sensitivity analysis." In: *Boreal Environment Research* 20.2.

Potter, Christopher S, James T Randerson, Christopher B Field, Pamela A Matson, Peter M Vitousek, Harold A Mooney, and Steven A Klooster (1993). "Terrestrial ecosystem production: a process model based on global satellite and surface data". In: *Global Biogeochemical Cycles* 7.4, pp. 811–841.

Sitch, Stephen et al. (2003). "Evaluation of ecosystem dynamics, plant geography and terrestrial carbon cycling in the LPJ dynamic global vegetation model". In: *Global Change Biology* 9.2, pp. 161–185.

Solomon, Susan (2007). *Climate change 2007-the physical science basis: Working group I contribution to the fourth assessment report of the IPCC*. Vol. 4. Cambridge University Press.

Annex

Tables

Table 2: Reference, minimum and maximum values of parameters.

name	def	min	max
soildepth	413.00	-	-
ThetaFC	0.45	-	-
ThetaPWP	0.12	-	-
tauDrainage	3.00	-	-
beta	0.75	0.2	2.5
tau	13.23	1	25
X[0]	-3.97	-20	20.1
S[max]	18.77	2.3	30
kappa	-0.13	-1	-0.001
gamma	0.03	0.000103	0.503
rho[P]	0.45	0	0.999
cmCO2	2000.00	-	-
ckappaCO2	0.40	-	-
alpha	0.32	1e-06	10
lambda	0.87	1e-04	1.2
chi	0.08	0	2.5
rho[ET]	0.54	0	0.9999
nu	0.27	1e-04	5
Meltcoef	1.20	-	-
I_0	0.33	-	-
CWmax	4.97	-	-
SnowThreshold	0.00	-	-
T_0	0.00	-	-
SWinit	200.00	-	-
CWinit	0.00	-	-
SOGinit	0.00	-	-
Sinit	20.00	-	-
t0	-999.00	-	-
tcrit	-999.00	-	-
tsumcrit	-999.00	-	-
sd_gpp	1.00	0.001	5
sd_et	1.00	0	10

Table 3: Parameters and initial conditions.

Name	Description	Unit
Parameters		
soildepth	Depth of the soil	mm
ThetaFC	Field capacity, above drainage occurs	mm
ThetaPWP	Wilting point	mm
tauDrainage	Fraction 1/tauDrainage of water above ThetaFC drains in a day	-
beta	Light use-efficiency	gC/mol
tau	season/temperature parameter	-
X[0]	Threshold above which the state of acclimation increases	°C
S[max]	Threshold at which the acclimation modifier reaches its maximum	°C
kappa	effect of VPD on GPP	-
gamma	Light modifier parameter for saturation with irradiance	$mol^{-1}m^{-2}$
rho[P]	Threshold for soil water effect on GPP in modifier	-
cmCO2	Mean effect of CO2 (ppm) on GPP	-
ckappaCO2	Change in CO2 effect on VPD	-
alpha	Parameter for transpiration, multiplier	$mm(gCm^{-2}kPa^{1-\lambda})^{-1}$
lambda	Light modifier parameter for saturation with irradiance	mol^{-1} m^{-2}
chi	Parameter for evaporation	mm mol^{-1}
rho[ET]	REW threshold when evaporation decreases	-
nu	Soil water correction for transpiration	-
Meltcoef	Melting coefficient of snow	-
I_0	Fraction of intercepted water	-
CWmax	Maximum storage for canopy (surfacial) water for free evaporation	mm
Initial conditions		
SWinit	Soil water at t = 0	mm
CWinit	Canopy water at t = 0	mm
SOGinit	Snow on ground at t = 0 (in water equivalents)	mm
Sinit	Temperature acclimation state at t = 0	°C

r-code

```
<<load packages>>=
library(Rpreles)
library(sensitivity)
library(randtoolbox)
library(BayesianTools)
library(lhs)
library(coda)
library(ExtDist)
library(timeDate)
library(xtable)
set.seed(123)
# kalibrierung: mcmc mit dream
@


\section{Sensitivity analysis}

<<preparing the parameters>>=
# loading the input data and observed data:
load("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/data/06-PrelesData/
    EddyCovarianceDataBorealSites.rdata")
# loading the parameter range values:
load("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/data/06-PrelesData/parameterRanges.rdata
    ")
pars_name = par$name
pars_ref = par$def
pars_index = which(par$min != par$max)
pars_max = par$max
pars_min = par$min
pars_range = list()
for(i in 1:length(pars_max)){
  pars_range[[i]] <- c(pars_min[i], pars_max[i])
}
names(pars_range) <- names(pars_ref) <- names(pars_min)
    <- names(pars_max) <- pars_name



par_l <- par
for (i in 1:length(par_l[,1])){
```

```
  if (par_l$min[i] == -999) par_l$min[i] <- "-"
  if (par_l$max[i] == -999) par_l$max[i] <- "-"
}

print(xtable(par_l), include.rownames=FALSE)
pars_latex <- xtable
save(pars_latex, file = "/home/simon/Dokumente/
    Umweltsystemmodellierung/Projektarbeit/pars_table.
    rdata")



pars_total_ref = c(pars_ref, mean(s1$PAR), mean(s1$TAir),
    mean(s1$VPD), mean(s1$Precip), mean(s1$CO2), mean(s1$
    fAPAR))
names(pars_total_ref) <- c(pars_name, "PAR", "TAir" , "
    VPD", "Precip", "CO2", "fAPAR")

pars_total_min <- c(pars_min, min(s1$PAR), min(s1$TAir),
    min(s1$VPD), min(s1$Precip), min(s1$CO2), min(s1$
    fAPAR))
pars_total_max <- c(pars_max, max(s1$PAR), max(s1$TAir),
    max(s1$VPD), max(s1$Precip), max(s1$CO2), max(s1$
    fAPAR))
names(pars_total_min) <- names(pars_total_max) <- c(pars_
    name, "PAR", "TAir" , "VPD", "Precip", "CO2", "fAPAR")
pars_total_index <- which(pars_total_min != pars_total_
    max)
  #c(pars_index, (length(pars_ref)+1):length(pars_total_
      ref))

# adding another parameter (for likelihood):
pars_ref[length(pars_ref)+1] <- 1
pars_min[length(pars_min)+1] <- 0
pars_max[length(pars_max)+1] <- 8
pars_index <- c(pars_index, 33)
@

<<run model function>>=
runModel = function(par){
    temp = pars_ref
    temp[pars_index] = par
    res <- PRELES(p = temp, PAR = s1$PAR, TAir = s1$TAir,
        VPD = s1$VPD, Precip = s1$Precip, CO2 = s1$CO2,
       fAPAR = s1$fAPAR, DOY = 1)
    res <- res$GPP
```

```
    return(res)
}
# run model with mean input values
runModelOnce = function(par, output = 1){
    temp = pars_ref
    temp[pars_index] = par
    res <- PRELES(p = temp, PAR = mean(s1$PAR), TAir =
        mean(s1$TAir), VPD = mean(s1$VPD), Precip = mean(
        s1$Precip), CO2 = mean(s1$CO2), fAPAR = mean(s1$
        fAPAR), DOY = 1)
    res <- res[[output]]
    return(res)
}
# rund model with parameter + inputvalues
runModel_total = function(par_total){
    temp = pars_total_ref
    temp[pars_total_index] = par_total
    res <- PRELES(p = temp[1:30], PAR = temp[32+1], TAir
        = temp[32+2], VPD = temp[32+3], Precip = temp
        [32+4], CO2 = temp[32+5], fAPAR = temp[32+6], DOY
        = 1)
    res <- res$GPP
    return(res)
}
@

<<paralle function>>=
parallelFun <- function(parMat){
  res <- apply(parMat, 1, runModel)
  if(! is.null(dim(res))) res = t(res) # to have results
      row-wise if multiple results are returned
  return(res)
}
parallelFun2 <- function(parMat){
  res <- apply(parMat, 1, runModel_total)
  if(! is.null(dim(res))) res = t(res) # to have results
      row-wise if multiple results are returned
  return(res)
}
@


\subsection{Local SA}
<<local sa>>=
gpp_ref <- runModelOnce(pars_ref[pars_index],1)
```

```
localSens <- function(deltap, parameter_ref, output_ref,
    return = "delta", output = 1){
  sens_delta <- array()
  sens_percent <- array()
  for (i in 1:length(parameter_ref)){
    temp <- parameter_ref
    temp[i] <- temp[i] * deltap
    output_var <- runModelOnce(temp, output)
    sens_delta[i] <- (output_var - output_ref) / (temp[i]
        -
                              parameter_ref[i])
    sens_percent[i] <- ((output_var - output_ref) /
       output_ref) / abs(1-deltap)
  }
  if (return == "delta") return(sens_delta)
  if (return == "percent") return(sens_percent)
}
gpp_sens_1.1 <- localSens(1.1, pars_ref[pars_index], gpp_
    ref, return = "percent", output = 1) # 1 for GPP
gpp_sens_0.9 <- localSens(0.9, pars_ref[pars_index], gpp_
    ref, return = "percent")

#plotting the results
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/local_sa.pdf")
plot(gpp_sens_1.1[1:12], type = "h", ylim = c(-1,1), lwd
    = 10, xaxt = "n", xlab = "")
axis(side = 1, at = 1:12,labels = names(pars_total_ref)[
    pars_total_index], tck = -0.01)
lines(seq(1.4,12.4,by=1),gpp_sens_0.9, type = "h", lwd =
    10, col ="grey")
dev.off()



#same for all variables, including input (boundary
    conditions):
gpp_ref <- runModelOnce(pars_ref[pars_index],1)
localSens_total <- function(deltap, parameter_ref, output
    _ref, return = "delta", output = 1){
  sens_delta <- array()
  sens_percent <- array()
  for (i in 1:length(parameter_ref)){
    temp <- parameter_ref
```

```
    temp[i] <- temp[i] * deltap
    output_var <- runModel_total(temp)
    sens_delta[i] <- (output_var - output_ref) / (temp[i]
        -
                              parameter_ref[i])
    sens_percent[i] <- ((output_var - output_ref) /
       output_ref) / abs(1-deltap)
  }
  if (return == "delta") return(sens_delta)
  if (return == "percent") return(sens_percent)
}


@
<<plot local SA>>=
gpp_sens_total_1.1 <- localSens_total(1.1, pars_total_ref
    [pars_total_index], gpp_ref, return = "percent") # 1
    for GPP
gpp_sens_total_0.9 <- localSens_total(0.9, pars_total_ref
    [pars_total_index], gpp_ref, return = "percent")

pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/local_sa.pdf")
plot(gpp_sens_total_1.1, type = "h", ylim = c(-1,1), lwd
    = 10, xaxt = "n", xlab = "", ylab = "%Out / %p")
axis(side = 1, at = 1:19,labels = names(pars_total_ref)[
    pars_total_index], tck = -0.01,las = 2)
lines(seq(1.4,19.4,by=1),gpp_sens_total_0.9, type = "h",
    lwd = 10, col ="grey")
dev.off()
@
<<delsa>>=
delsaOut <- delsa(model = parallelFun, par.ranges = pars_
    range[pars_index], method = "sobol", samples = 300)
plot(delsaOut)
@



\subsection{Global SA}

<<run morris 1, without boundary conditions>>=
morrisOut_300 <- morris(model = parallelFun, factors =
    pars_name[pars_index], r = 300,
                        design = list(type = "oat", levels =
                            10, grid.jump = 5),
```

```
                        binf = pars_min[pars_index], bsup =
                            pars_max[pars_index])
save(morrisOut_300, file = "/home/simon/Dokumente/
    Umweltsystemmodellierung/Projektarbeit/morrisOut_300.
    RData")
load("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/morrisOut_300.RData")
plot(morrisOut_300)

morrisOut_1000 <- morris(model = parallelFun, factors =
    pars_name[pars_index], r = 1000,
                        design = list(type = "oat", levels =
                             10, grid.jump = 5),
                        binf = pars_min[pars_index], bsup =
                            pars_max[pars_index])
save(morrisOut_1000, file = "/home/simon/Dokumente/
    Umweltsystemmodellierung/Projektarbeit/morrisOut_1000.
    RData")
load("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/morrisOut_1000.RData")
plot(morrisOut_1000)
print(morrisOut_1000)

@

<<morris 2 including boundary conditions>>=
# same for all pars (also input variables)
morrisOut_300_total <- morris(model = parallelFun2,
    factors = names(pars_total_ref)[pars_total_index], r =
     300,
                        design = list(type = "oat", levels =
                             10, grid.jump = 5),
                        binf = pars_total_min[pars_total_
                            index], bsup = pars_total_max[
                            pars_total_index])
print(morrisOut_300_total)
plot(morrisOut_300_total)

morrisOut_1000_total <- morris(model = parallelFun2,
    factors = names(pars_total_ref)[pars_total_index], r =
     1000,
                        design = list(type = "oat", levels =
                             10, grid.jump = 5),
                        binf = pars_total_min[pars_total_
                            index], bsup = pars_total_max[
                            pars_total_index])
```

```
print(morrisOut_1000_total)
plot(morrisOut_1000_total)

morrisOut_5000_total <- morris(model = parallelFun2,
    factors = names(pars_total_ref)[pars_total_index], r =
     5000,
                        design = list(type = "oat", levels =
                             10, grid.jump = 5),
                        binf = pars_total_min[pars_total_
                            index], bsup = pars_total_max[
                            pars_total_index])
print(morrisOut_5000_total)
plot(morrisOut_5000_total)

@

<<plot morris>>=
#plot morris
mu <- apply(morrisOut_1000_total$ee, 2, mean)
mu.star <- apply(morrisOut_1000_total$ee, 2, function(x)
    mean(abs(x)))
sigma <- apply(morrisOut_1000_total$ee, 2, sd)

pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/local_sa.pdf")
par(mfrow=c(1,1))
plot(gpp_sens_total_1.1, type = "h", ylim = c(-1,1), lwd
    = 10, xaxt = "n", xlab = "", ylab = "%Out / %p", main
    = "Local SA")
axis(side = 1, at = 1:19,labels = names(pars_total_ref)[
    pars_total_index], tck = -0.01,las = 2)
lines(seq(1.4,19.4,by=1),gpp_sens_total_0.9, type = "h",
    lwd = 10, col ="grey")
dev.off()
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/global_sa.pdf")
plot(mu.star, xaxt = "n", xlab = "", type = "h", lwd =
    10, main = "Morris screening")
axis(side = 1, at = 1:19,labels = names(pars_total_ref)[
    pars_total_index], tck = -0.01,las = 2)
dev.off()
@

<<defining influential pars>>=
pars2calibrate <- c(which(names(pars_ref) == "gamma"),
    which(names(pars_ref) == "S[max]"), which(names(pars_
```

```
    ref) == "tau"), which(names(pars_ref) == "X[0]"),
    which(names(pars_ref) == "beta"),which(names(pars_ref)
     == "sd_gpp"),33)
@

\section{Parameter calibration}

<<likelihood function 1>>=
likelihood <- function(x, sum = TRUE){
  x <- createMixWithDefaults(x, pars_ref, pars2calibrate)
       # ersetzte fehlende parameter mit den default-
      werten
   predicted <- PRELES(p = x[1:30], PAR = s1$PAR, TAir =
      s1$TAir, VPD = s1$VPD, Precip = s1$Precip, CO2 = s1$
      CO2, fAPAR = s1$fAPAR, DOY = s1$DOY)
   diff <- c(predicted$GPP - s1$GPPobs)
   #llValues <- sum(dnorm(diff, sd = x[31]+x[33]*predicted
      $GPP, log=T))
   llValues <- dnorm(diff, sd = x[31], log = T)
   if(sum == FALSE){return(llValues)}
   else {return(sum(llValues))}
}
@
<<create prior 1>>=
prior <- createUniformPrior(lower = pars_min[
    pars2calibrate], upper = pars_max[pars2calibrate],
    best = pars_ref[pars2calibrate])
@
<<bayesian setup 1>>=
bayesianSetup <- createBayesianSetup(likelihood, prior,
    names = pars_name[pars2calibrate])
bayesianSetup$prior$density(pars_ref[pars2calibrate])
bayesianSetup$likelihood$density(pars_ref[pars2calibrate
    ])
bayesianSetup$posterior$density(pars_ref[pars2calibrate])
@

Now sensitivity analysis on likelihood:
<<SA on likelihood>>=
morrisOut_ll <- morris(model = bayesianSetup$posterior$
    density, factors = pars_name[pars_index], r = 2000,
    design = list(type = "oat", levels = 5, grid.jump = 3)
    , binf = pars_min[pars_index], bsup = pars_max[pars_
    index], scale = TRUE)
save(morrisOut_ll, file = "/home/simon/Dokumente/
    Umweltsystemmodellierung/Projektarbeit/morrisOut_ll.
```

```
    RData")
load("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/morrisOut_ll.RData")
plot(morrisOut_ll)
@
<<SA on likelihood2>>=
morrisOut_ll2 <- morris(model = bayesianSetup$posterior$
    density, factors = pars_name[pars2calibrate], r =
    2000, design = list(type = "oat", levels = 5, grid.
    jump = 3), binf = pars_min[pars2calibrate], bsup =
    pars_max[pars2calibrate], scale = TRUE)
save(morrisOut_ll2, file = "/home/simon/Dokumente/
    Umweltsystemmodellierung/Projektarbeit/morrisOut_ll2.
    RData")
load("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/morrisOut_ll2.RData")
plot(morrisOut_ll2)
@


<<MCMC 1>>=
settings <- list(iterations = 100000, nrChains = 3)
mcmcOut3 <- runMCMC(bayesianSetup = bayesianSetup,
    sampler = "DEzs", settings = settings)
plot(mcmcOut3)
save(mcmcOut3, file = "/home/simon/Dokumente/
    Umweltsystemmodellierung/Projektarbeit/mcmcOut3.RData"
    )
load("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/mcmcOut3.RData")
tracePlot(mcmcOut3, parametersOnly = TRUE, start = 1)
gelmanDiagnostics(mcmcOut3, start = 15000, plot = T)

settings <- list(iterations = 200000, nrChains = 3)
mcmcOut_200000 <- runMCMC(bayesianSetup = bayesianSetup,
    sampler = "DEzs", settings = settings)
plot(mcmcOut_200000)
save(mcmcOut_200000, file = "/home/simon/Dokumente/
    Umweltsystemmodellierung/Projektarbeit/mcmcOut_200000.
    RData")

@

Interpreting the mcmc results:
<<MCMC 1 plots>>=
par(mfrow=c(1,1))
```

```
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/mcmc_marg.pdf")
marginalPlot(mcmcOut_200000, start = 20000, scale = cbind
    (pars_min[pars2calibrate], pars_max[pars2calibrate]),
    best = pars_ref[pars2calibrate]) # skalierung
dev.off()
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/mcmc_corr.pdf")
correlationPlot(mcmcOut_200000, parametersOnly = TRUE,
    start = 20000) # correlation plot
dev.off()
@
strong correlation of gamma and beta, as well as s[max]
    and x[0]



\section{Validation}
<<Simulation with MCMC 1>>=
Sample <- getSample(mcmcOut_200000,thin=10,start=0.5*
    settings$iterations/settings$nrChains,coda=T)
p <- pars_ref
nSample <- nrow(Sample[[1]])
GPPerror <- GPPsim <- matrix(0, nrow = nSample, ncol =
    length(s1$GPPobs))

for (i in 1:nrow(Sample[[1]])){
  p[pars2calibrate] <- Sample[[1]][i, ]
   preles_s1 <- PRELES(DOY=1,PAR=s1$PAR, TAir=s1$TAir, VPD
       =s1$VPD,
                           Precip=s1$Precip, CO2=s1$CO2, fAPAR
                               =s1$fAPAR, p=p[1:30])
  #GPPerror[i,] <- sample(c(-1,1), 1) * rexp(length(
      preles_s1$GPP),rate = 1/(p[31]*0.1*preles_s1$GPP))
  GPPsim[i, ] <- preles_s1$GPP
  #GPPerror[i, ] <- rnorm(length(preles_s1$GPP), mean =
      0, sd = p[31] + p[33]*preles_s1$GPP)
  GPPerror[i, ] <- rnorm(length(preles_s1$GPP),sd = p
      [31])
}
@

<<p value 1>>=
p_value <- numeric(length(s1$GPPobs))
for (i in seq(p_value)) {
  p_value[i] <- mean(GPPsim[, i] + GPPerror[, i] < s1$
```

```
      GPPobs[i])
}
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/p_value_plot.pdf")
plot(p_value, xlab = "time [days]")
dev.off()
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/p_value_hist.pdf")
hist(p_value, main = '') # noch drehen!!!
dev.off()

@
In the p-value there can be observed a clear pattern over
     time. During winter the

<<heteroscedasticity 1>>=
med_pred <- apply(GPPsim, 2, median)
plot(med_pred, p_value)
@
<<res ~ preds 1>>=
med_resid <- apply(s1$GPPobs - t(GPPsim), 1, median)
par(mfrow = c(1,1))
plot(med_pred, med_resid)
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/res_obs.pdf")
plot(s1$GPPobs, med_resid, xlab = "Observed Values of GPP
    ", ylab = "median of Residuals")
dev.off()

@
<<res ~ predictor variables 1>>=
predictors <- c("PAR", "TAir", "VPD", "Precip")
par(mfrow = c(2, 2))
for (i in predictors) {
  plot(s1[[i]], med_resid, xlab = i, xlim = range(s1[[i
      ]][!is.na(med_resid)]))
}
@



\section{Parameter calibration 2}
<<fixing all this>>=
# define 2 pars for quadratic influence of GPP on sigma
pars2 <- data.frame()
pars2 <- cbind(pars_ref, pars_min, pars_max)
```

```
a <- c(1,0,5)
b <- c(1,0,5)
pars2 <- as.data.frame(rbind(pars2, a, b))
rownames(pars2)[31] <- "c"
pars2calibrate2 <- c(pars2calibrate, 33,34)
@
<<likelihood 2>>=
likelihood2 <- function(pValues) {
  p <- pars2$pars_ref
  p[pars2calibrate2] <- pValues
  predicted <- PRELES(DOY=s1$DOY,PAR=s1$PAR, TAir=s1$TAir,
      VPD=s1$VPD, Precip=s1$Precip ,CO2=s1$CO2,fAPAR=s1$
      fAPAR,LOGFLAG=0,p=p[1:30], control=1)
  diff_GPP <- predicted$GPP - s1$GPPobs
  #llvalues <- sum(dexp(abs(diff_GPP), rate = 1/(p[31]*
      predicted$GPP), log=T))
  llvalues <- sum(dnorm(diff_GPP, sd = p[31]+p[33]*
      predicted$GPP+p[34]*(predicted$GPP)^2,log=T))
  # llvalues <- sum(dnorm(diff_GPP, sd = p[31], log=T))
  return(llvalues)
}
@

<<prior 2>>=
prior2 <- createUniformPrior(lower = pars2$pars_min[
    pars2calibrate2], upper = pars2$pars_max[
    pars2calibrate2], best = pars2$pars_ref[
    pars2calibrate2])
@

<<bayesian setup 2>>=
bayesianSetup2 <- createBayesianSetup(likelihood2, prior2
    , names = rownames(pars2)[pars2calibrate2])
@

<<MCMC 2>>=
settings2 <- list(iterations = 500000, nrChains = 3)
mcmcOut22 <- runMCMC(bayesianSetup2, sampler="DEzs",
    settings = settings2)
save(mcmcOut22, file = "/home/simon/Dokumente/
    Umweltsystemmodellierung/Projektarbeit/mcmcOut22.RData
    ")
load("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/mcmcOut2.RData")
```

```
pars_sim <- summary(mcmcOut22)$statistics
pars_sim
table4tex <- cbind(pars2[pars2calibrate2,],(pars_sim[,1])
    )
colnames(table4tex)[4] <- "calibrated (mean)"
print(xtable(table4tex), include.rownames=T)
save(table4tex, file = "/home/simon/Dokumente/
    Umweltsystemmodellierung/Projektarbeit/pars_values.
    rdata")

pars2
@

<<MCMC plots 2>>=
tracePlot(mcmcOut22, parametersOnly = TRUE, start = 1)
gelmanDiagnostics(mcmcOut2, start = 15000, plot = T)
par(mfrow=c(1,1))
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/mcmc_marg2.pdf")
marginalPlot(mcmcOut2, start = 15000, scale = cbind(pars2
    $pars_min[pars2calibrate2],pars2$pars_max[
    pars2calibrate2]), best = pars_ref[pars2calibrate]) #
    skalierung
dev.off()
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/mcmc_corr2.pdf")
correlationPlot(mcmcOut2, parametersOnly = TRUE, start =
    20000) # correlation plot
dev.off()
@


\section{Validation 2}

<<Simulation with MCMC 2>>=
Sample2 <- getSample(mcmcOut2,thin=10,start=0.2*settings2
    $iterations/settings2$nrChains,coda=T)
p <- pars2$pars_ref
nSample <- nrow(Sample2[[1]])
GPPerror2 <- GPPsim2 <- matrix(0, nrow = nSample, ncol =
    length(s1$GPPobs))

for (i in 1:nrow(Sample2[[1]])){
  p[pars2calibrate2] <- Sample2[[1]][i,]
   predicted <- PRELES(DOY=1,PAR=s1$PAR, TAir=s1$TAir, VPD
      =s1$VPD,
```

```
                                Precip=s1$Precip, CO2=s1$CO2, fAPAR
                                    =s1$fAPAR, p=p[1:30])
  #GPPerror[i,] <- sample(c(-1,1), 1) * rexp(length(
      preles_s1$GPP),rate = 1/(p[31]*0.1*preles_s1$GPP))
  GPPsim2[i, ] <- predicted$GPP
  GPPerror2[i, ] <- rnorm(length(preles_s1$GPP), sd = p
      [31]+p[33]*predicted$GPP+p[34]*(predicted$GPP)^2)
  # error <- rnorm(length(preles_s1$GPP),sd = p[31]+p[32]
      *)
}
@

<<p value 2>>=
p_value2 <- numeric(length(s1$GPPobs))
for (i in seq(p_value2)) {
  p_value2[i] <- mean(GPPsim2[, i] + GPPerror2[, i] < s1$
      GPPobs[i])
}
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/p_value_plot2.pdf")
plot(p_value2, xlab = "time [days]", ylab = "p-values")
dev.off()
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/p_value_hist2.pdf")
hist(p_value2, xlab = "p-values", main = '') # noch
    drehen!!!
dev.off()

@

<<heteroscedasticity>>=
med_pred2 <- apply(GPPsim2, 2, median)
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/p_value_pred.pdf")
plot(med_pred2, p_value2, ylab = "p-values", xlab = "
    Median of GPP prediction [gC m^-2 day^-1]",
      main = "p-value vs. predicted",pch = 20, lwd = 0.4)
dev.off()
#plot(med_pred, p_value) # for comparing
@
<<res ~ preds>>=
med_resid2 <- apply(s1$GPPobs - t(GPPsim2), 1, median)
par(mfrow = c(1,1))
plot(med_pred2, med_resid2)
```

```
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/res_obs.pdf")
plot(s1$GPPobs, med_resid2, xlab = "Observed Values of
    GPP [gC m^-2 day^-1]",
      ylab = "median of Residuals", pch = 20, lwd = 0.4,
          main = "Residuals vs. Observed")
dev.off()

@
<<res ~ predictor variables>>=
predictors <- c("PAR", "TAir", "VPD", "Precip")
par(mfrow = c(2, 2))
for (i in predictors) {
  plot(s1[[i]], p_value2, xlab = i, xlim = range(s1[[i]][
      !is.na(med_resid)]))
}
@

<<res ~ obs>>=

fm = lm(s1$GPPobs ~ med_pred2)
fm2 <- lm(s1$GPPobs ~ med_pred2 + I(med_pred2^2))
summary(fm)
summary(fm2)
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/pred_obs.pdf")
plot(med_pred2, s1$GPPobs, xlab = "Predicted GPP [gC m^-2
     day^-1]", ylab = "Observed GPP [gC m^-2 day^-1]", pch
     = 20, lwd = 0.4, col = "darkgrey", main = "Predicted
    vs. Observed")
curve(fm$coefficients[2] * x + fm$coefficients[1], from =
     min(med_pred2), to = max(med_pred2), add = T, lwd =
    2)
legend("topleft", legend = c("GPP_obs = f(GPP_pred) = GPP
    _pred","","r   = 0.96"), bty="n")
dev.off()

@


<<too extreme res?>>=
qqnorm(med_resid2)
@

<<autocorrelation>>=
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
```

```
    Projektarbeit/autocorr2.pdf")
ac <- acf(ts(p_value2), na.action = na.pass, main = "
    Autocorrelation of p-values")
dev.off()
@


<<log normal>>=
likelihood3 <- function(pValues) {
  p <- pars2$pars_ref
  p[pars2calibrate2] <- pValues
  predicted <- PRELES(DOY=s1$DOY,PAR=s1$PAR, TAir=s1$TAir,
      VPD=s1$VPD, Precip=s1$Precip ,CO2=s1$CO2,fAPAR=s1$
      fAPAR,LOGFLAG=0,p=p[1:30], control=1)
  diff_GPP <- predicted$GPP - s1$GPPobs
  #llvalues <- sum(dexp(abs(diff_GPP), rate = 1/(p[31]*
      predicted$GPP), log=T))
  llvalues <- sum(dlnorm(s1$GPPobs, meanlog = exp(
      predicted$GPP+((p[31]+p[33]*predicted$GPP)^2)/2),
      sdlog = exp(p[31]+p[33]*predicted$GPP)))
  # llvalues <- sum(dnorm(diff_GPP, sd = p[31], log=T))
  return(llvalues)
}
bayesianSetup3 <- createBayesianSetup(likelihood3, prior2
    , names = rownames(pars2)[pars2calibrate2])

settings <- list(iterations = 50000, nrChains = 3)
mcmcOut3 <- runMCMC(bayesianSetup3, sampler="DEzs",
    settings = settings)

tracePlot(mcmcOut3, parametersOnly = TRUE, start = 1)

Sample <- getSample(mcmcOut3,thin=10,start=0.2*settings$
    iterations/settings$nrChains,coda=T)
p <- pars2$pars_ref
nSample <- nrow(Sample[[1]])
GPPerror <- GPPsim <- matrix(0, nrow = nSample, ncol =
    length(s1$GPPobs))

for (i in 1:nrow(Sample[[1]])){
  p[pars2calibrate2] <- Sample[[1]][i, ]
  predicted <- PRELES(DOY=1,PAR=s1$PAR, TAir=s1$TAir, VPD
      =s1$VPD,
                      Precip=s1$Precip, CO2=s1$CO2, fAPAR
                          =s1$fAPAR, p=p[1:30])
  #GPPerror[i,] <- sample(c(-1,1), 1) * rexp(length(
```

```
      preles_s1$GPP), rate = 1/(p[31]*0.1*preles_s1$GPP))
  GPPsim[i, ] <- preles_s1$GPP
  GPPerror[i, ] <- rlnorm(length(preles_s1$GPP), meanlog
      = exp(predicted$GPP+((p[31]+p[33]*predicted$GPP)^2)/
      2), sdlog = exp(p[31]+p[33]*predicted$GPP))
  # error <- rnorm(length(preles_s1$GPP),sd = p[31]+p[32]
      *)
}

@

<<get quantiles>>=
quant_func <- function(x){
  res <- quantile(x, probs = c(0.025, 0.975))
  return(res)
}

quantiles_conf2 <- apply(GPPsim2,2, quant_func)
max_pred2 <- apply(GPPsim2 + GPPerror2,2, max)
min_pred2 <- apply(GPPsim2 + GPPerror2,2, min)


par(mfrow = c(1,1))
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/predictions.pdf")

plot(s1$GPPobs,col=2, xlab='',ylab='tC ha-1', main='GPP',
    ylim=c(-3,22), type = "n")
polygon(c(seq(1,730), rev(seq(1,730))), c(min_pred2, rev(
    max_pred2)), col = "coral1", border = "coral1")
polygon(c(seq(1,730), rev(seq(1,730))), c(quantiles_conf2
    [2,], rev(quantiles_conf2[1,])), col = "deepskyblue3",
     border = "deepskyblue3")
points(s1$GPPobs,col="grey2", pch = 4, lwd = 0.1)
abline(h = 0, lty = 3)
legend("topleft", legend = c("","","Observed GPP"), bty =
     "n", pch = c("", "", "x"),
        fill = c("white","white", "white"), border = c("
            white", "white", "white"))
legend("topleft", legend = c("95 % confidence interval","
    Prediction uncertainty"), bty="n", fill = c("
    deepskyblue3", "coral2"), border = c("deepskyblue3", "
    coral2"))

dev.off()
```

```
@

<<p value>>=
p_value2 <- numeric(length(s1$GPPobs))
for (i in seq(p_value2)) {
  p_value2[i] <- mean(GPPsim[, i] + GPPerror[, i] < s1$
      GPPobs[i])
}
plot(p_value2)
hist(p_value2)
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/p_value_plot2.pdf")
plot(p_value2, xlab = "time [days]")
dev.off()
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/p_value_hist2.pdf")
hist(p_value2, main = '') # noch drehen!!!
dev.off()
@
<<gpp plot>>=
par(mfrow = c(1,1))
plot(s1$GPPobs,col=2, xlab='',ylab='tC ha-1', main='GPP',
    ylim=c(-3,15), type = "n")
for (i in 1:nrow(GPPsim2)) {
#  lines(quantiles[2,i], col='light green')
  lines(GPPsim2[i, ] + GPPerror2[i, ], col='light green')
  lines(GPPsim2[i, ], col='dark green')
}
points(s1$GPPobs,col=2)
abline(h = 0, lty = 3)
@



\section{Prediction Uncertainty}
<<get uncertainty on prediction>>=

quant_func <- function(x){
  res <- quantile(x, probs = c(0.025, 0.975))
  return(res)
}
quantile(GPPsim[,1], probs = c(0.025, 0.975))
quantiles <- apply(GPPsim,2, quant_func)
plot(quantiles[1,])
quantiles[,1]
polygon(quantiles[2,])
```

```
polygon(quantiles[1,])
@
<<gpp plot>>=
pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/uncert_pred.pdf")
plot(s1$GPPobs,col=2, xlab='',ylab='tC ha-1', main='GPP',
    ylim=c(-3,15), type = "n")
for (i in 1:nrow(GPPsim)) {
#  lines(quantiles[2,i], col='light green')
  lines(GPPsim[i, ] + GPPerror[i, ], col='light green')
  lines(GPPsim[i, ], col='dark green')
}
points(s1$GPPobs,col=2)
abline(h = 0, lty = 3)
dev.off()



@

\section{Prediction on climate change}
<<making prediction>>=
CO2_cc = mean(c(532.00 ,  501.00   ,567.00 ,  532.00
    ,  488.00 ,  478.00   ,538.00 ,  535.00  ,  472.00
    ,  486.00  ,  508.00   , 509.00))
CO2_cc_vec = rep(CO2_cc, times = 730)

p <- pars2$pars_ref
nSample <- nrow(Sample2[[1]])
GPPerror_cc <- GPPsim_cc <- matrix(0, nrow = nSample,
    ncol = length(s1$GPPobs))
for (i in 1:nrow(Sample2[[1]])){
  p[pars2calibrate2] <- Sample2[[1]][i, ]
  predicted <- PRELES(DOY=1,PAR=s1$PAR, TAir=s1$TAir+5,
      VPD=s1$VPD,
                      Precip=s1$Precip*1.3, CO2=CO2_cc_
                          vec, fAPAR=s1$fAPAR, p=p[1:30])
  #GPPerror[i,] <- sample(c(-1,1), 1) * rexp(length(
      preles_s1$GPP),rate = 1/(p[31]*0.1*preles_s1$GPP))
  GPPsim_cc[i, ] <- predicted$GPP
  GPPerror_cc[i, ] <- rnorm(length(predicted$GPP), sd = p
      [31]+p[33]*predicted$GPP+p[34]*(predicted$GPP)^2)
  # error <- rnorm(length(preles_s1$GPP),sd = p[31]+p[32]
      *)
}
```

```
@

<<plotting climate change>>=
quantiles_conf_cc <- apply(GPPsim_cc,2, quant_func)
max_pred_cc <- apply(GPPsim_cc + GPPerror_cc,2, max)
min_pred_cc <- apply(GPPsim_cc + GPPerror_cc,2, min)


pdf("/home/simon/Dokumente/Umweltsystemmodellierung/
    Projektarbeit/uncert_pred_cc.pdf")
plot(s1$GPPobs,col=2, xlab='',ylab='tC ha-1', main='GPP
    under climate change',ylim=c(-3,22), type = "n")
polygon(c(seq(1,730), rev(seq(1,730))), c(min_pred_cc,
    rev(max_pred_cc)), col = "coral1", border = "coral1")
polygon(c(seq(1,730), rev(seq(1,730))), c(quantiles_conf_
    cc[2,], rev(quantiles_conf_cc[1,])), col = "
    deepskyblue3", border = "deepskyblue3")
abline(h = 0, lty = 3)
dev.off()
@

<<increase of gpp>>=
mean(GPPsim_cc)/mean(GPPsim2)
summer_ind <- c(128:310, (365+128):(365+310))
winter_ind <- c(0:127, 311:(365 + 127), (365+311):730)

mean(GPPsim_cc[,summer_ind])/mean(GPPsim2[,summer_ind])
mean(GPPsim_cc[,winter_ind])/mean(GPPsim2[,winter_ind])


increase <- GPPsim_cc - GPPsim2
increase_err <- GPPerror_cc - GPPerror2
quantiles_conf_dif <- apply(increase,2, quant_func)
max_pred_dif <- apply(increase_err,2, max)
min_pred_dif <- apply(increase_err,2, min)
plot(s1$GPPobs,col=2, xlab='',ylab='tC ha-1', main='GPP
    under climate change',ylim=c(-22,22), type = "n")
polygon(c(seq(1,730), rev(seq(1,730))), c(min_pred_dif,
    rev(max_pred_dif)), col = "coral1", border = "coral1")
polygon(c(seq(1,730), rev(seq(1,730))), c(quantiles_conf_
    dif[2,], rev(quantiles_conf_dif[1,])), col = "
    deepskyblue3", border = "deepskyblue3")
abline(h = 0, lty = 3)
```

@

end{document}