2.1 Linking Models and Reality:
Reification Strategies for a Rainfall-Runoff Model.

Ian Vernon*

Dept. Mathematical Sciences, University of Durham, UK

* work done in collaboration with Michael Goldstein, Allan Seheult, Leanna House, (Dept. Mathematical Sciences, Durham University), Prof Keith Beven (Lancaster University) Funding: MUCM and EPSRC.

Overview

- Reification Strategies.
- The Rainfall-runoff model.
- Dynamic Reification.
- Modular Reification.

Reification Strategies.

- Goal is to reify the Rainfall-runoff model.

Reification Strategies.

- Goal is to reify the Rainfall-runoff model.
- Several Reification Strategies are available:
  - Direct Emulation and Reification (see Leanna's internal report).

Reification Strategies.

- Goal is to reify the Rainfall-runoff model.
- Several Reification Strategies are available:
  - Direct Emulation and Reification (see Leanna's internal report).
  - Dynamic Emulation and Reification.

Reification Strategies.

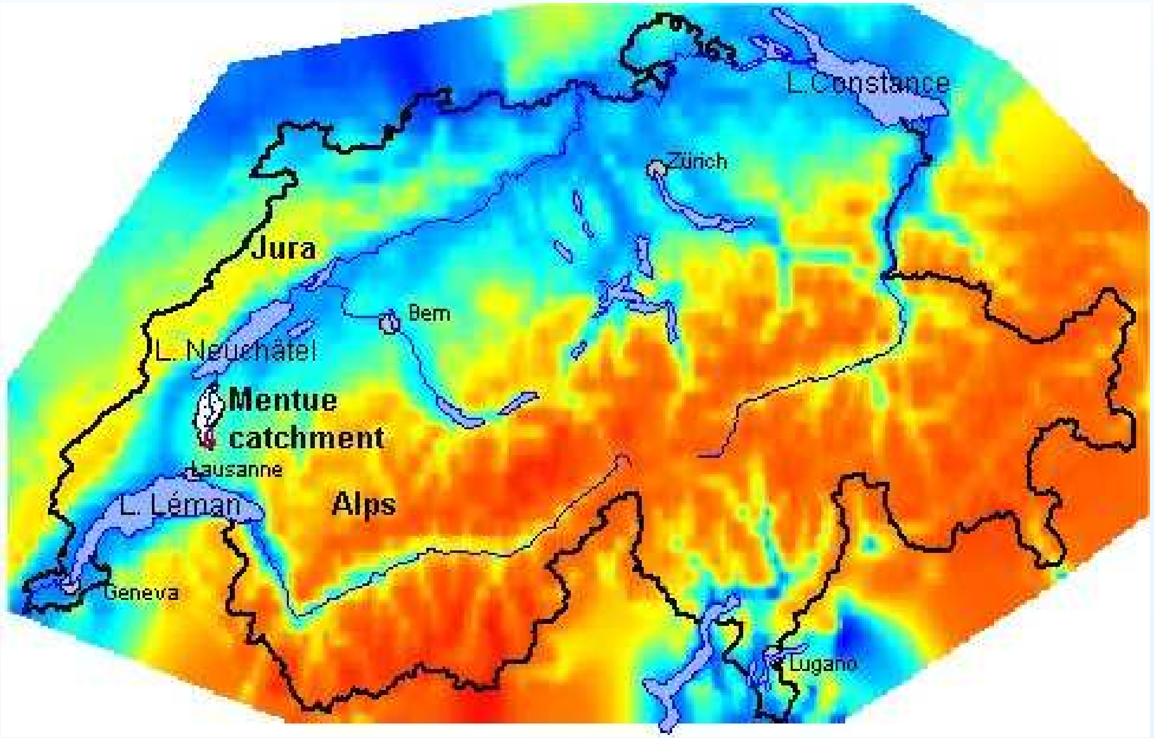
- Goal is to reify the Rainfall-runoff model.
- Several Reification Strategies are available:
  - Direct Emulation and Reification (see Leanna's internal report).
  - Dynamic Emulation and Reification.
  - Modular Emulation and Reification.

Reification Strategies.

- Goal is to reify the Rainfall-runoff model.
- Several Reification Strategies are available:
  - Direct Emulation and Reification (see Leanna's internal report).
  - Dynamic Emulation and Reification.
  - Modular Emulation and Reification.
- The further down the list we go, the easier the Reification, but the harder the Emulation.

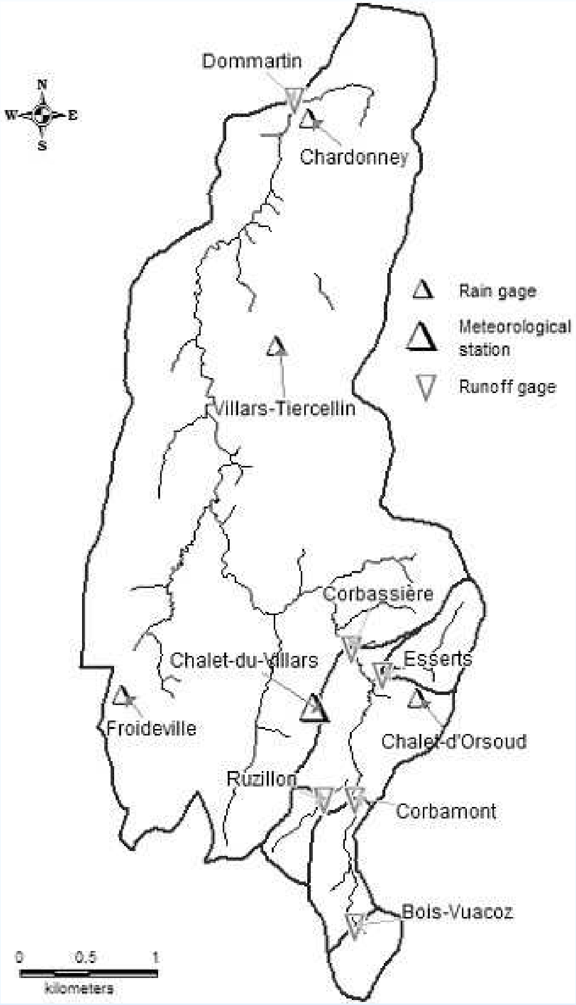
The Flood-Runoff Model of the Haute-Mentue catchment.

- West part of Switzerland, 15 km north of Lausanne, in the canton of Vaud.



The Flood-Runoff Model of the Haute-Mentue catchment.

- West part of Switzerland, 15 km north of Lausanne, in the canton of Vaud.



The Flood-Runoff Model of the Haute-Mentue catchment.

- Models water discharged through streams over an 839 hour period in the Haute-Mentue research catchment (Switzerland).

The Flood-Runoff Model of the Haute-Mentue catchment.

- Models water discharged through streams over an 839 hour period in the Haute-Mentue research catchment (Switzerland).

- Rainfall flows into one of 3 compartments where it is stored for a fast or slow amount of time, before being discharged into the streams.

The Flood-Runoff Model of the Haute-Mentue catchment.

- Models water discharged through streams over an 839 hour period in the Haute-Mentue research catchment (Switzerland).
- Rainfall flows into one of 3 compartments where it is stored for a fast or slow amount of time, before being discharged into the streams.
- We have two or three compartment versions of the model.

The Flood-Runoff Model of the Haute-Mentue catchment.

- Models water discharged through streams over an 839 hour period in the Haute-Mentue research catchment (Switzerland).

- Rainfall flows into one of 3 compartments where it is stored for a fast or slow amount of time, before being discharged into the streams.

- We have two or three compartment versions of the model.

- There are 6 inputs per compartment of the model $a_i, b_i, k_i, p_i, cs_i, cf_i$, and one constraint giving 17 inputs x_i for the 3 compartment model.

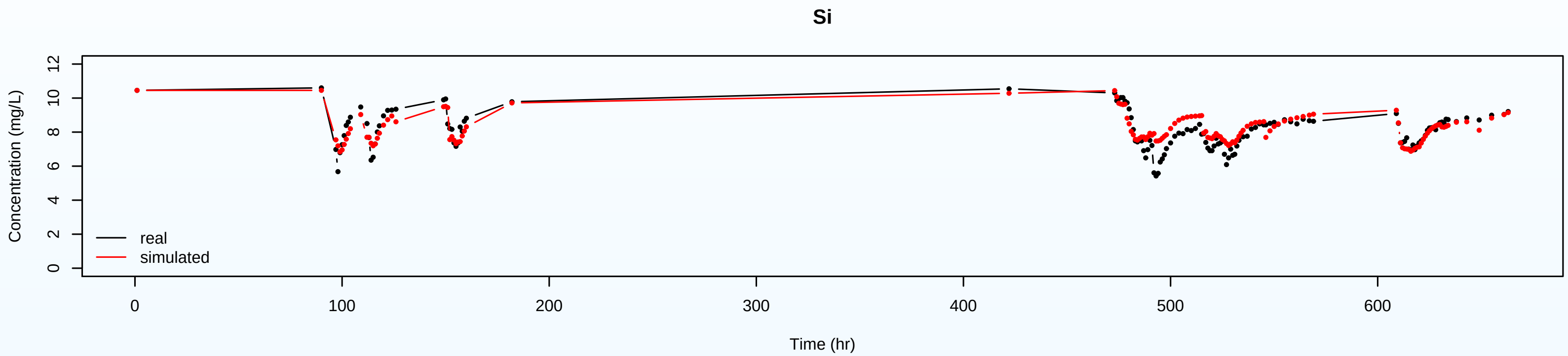
The Flood-Runoff Model of the Haute-Mentue catchment.

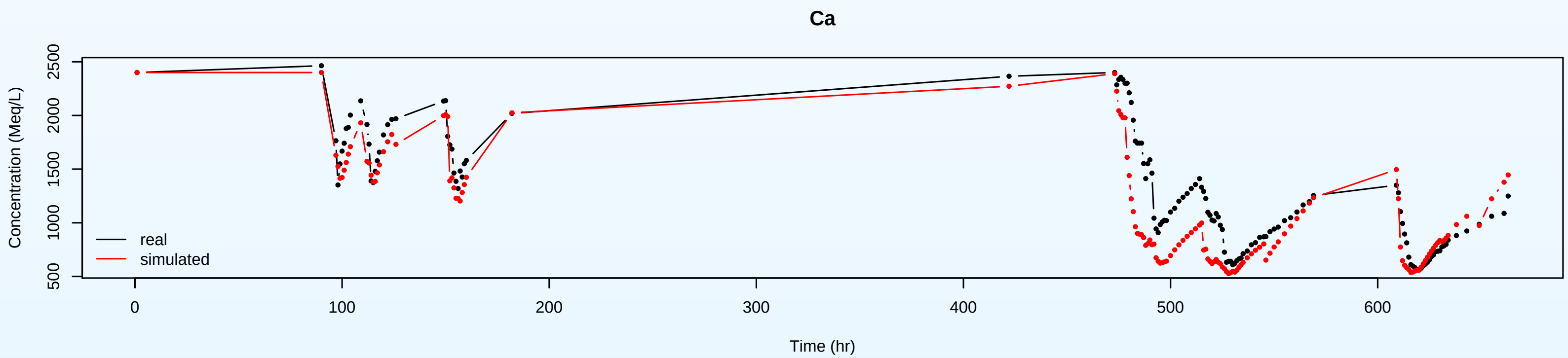
- Models water discharged through streams over an 839 hour period in the Haute-Mentue research catchment (Switzerland).
- Rainfall flows into one of 3 compartments where it is stored for a fast or slow amount of time, before being discharged into the streams.
- We have two or three compartment versions of the model.
- There are 6 inputs per compartment of the model $a_i, b_i, k_i, p_i, cs_i, cf_i$, and one constraint giving 17 inputs x_i for the 3 compartment model.
- Outputs are 3 time series representing the water discharged (PD), and the concentration of two tracer chemicals (Ca and Si) at each hour.

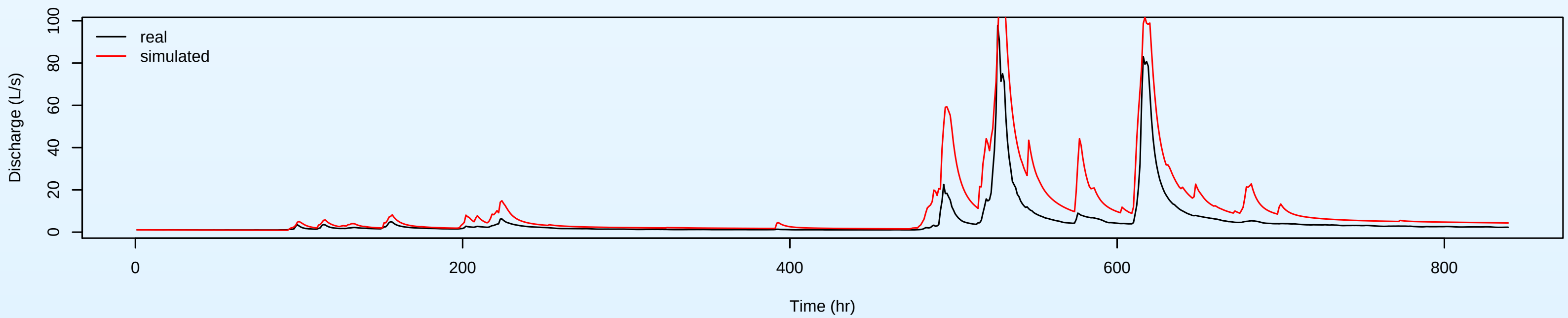
The Flood-Runoff Model of the Haute-Mentue catchment.

- Models water discharged through streams over an 839 hour period in the Haute-Mentue research catchment (Switzerland).
- Rainfall flows into one of 3 compartments where it is stored for a fast or slow amount of time, before being discharged into the streams.
- We have two or three compartment versions of the model.
- There are 6 inputs per compartment of the model $a_i, b_i, k_i, p_i, cs_i, cf_i$, and one constraint giving 17 inputs x_i for the 3 compartment model.
- Outputs are 3 time series representing the water discharged (PD), and the concentration of two tracer chemicals (Ca and Si) at each hour.
- Model takes measured rainfall rates $rain(t)$ and assumed evaporation rates $AET(t)$ as forcing functions.

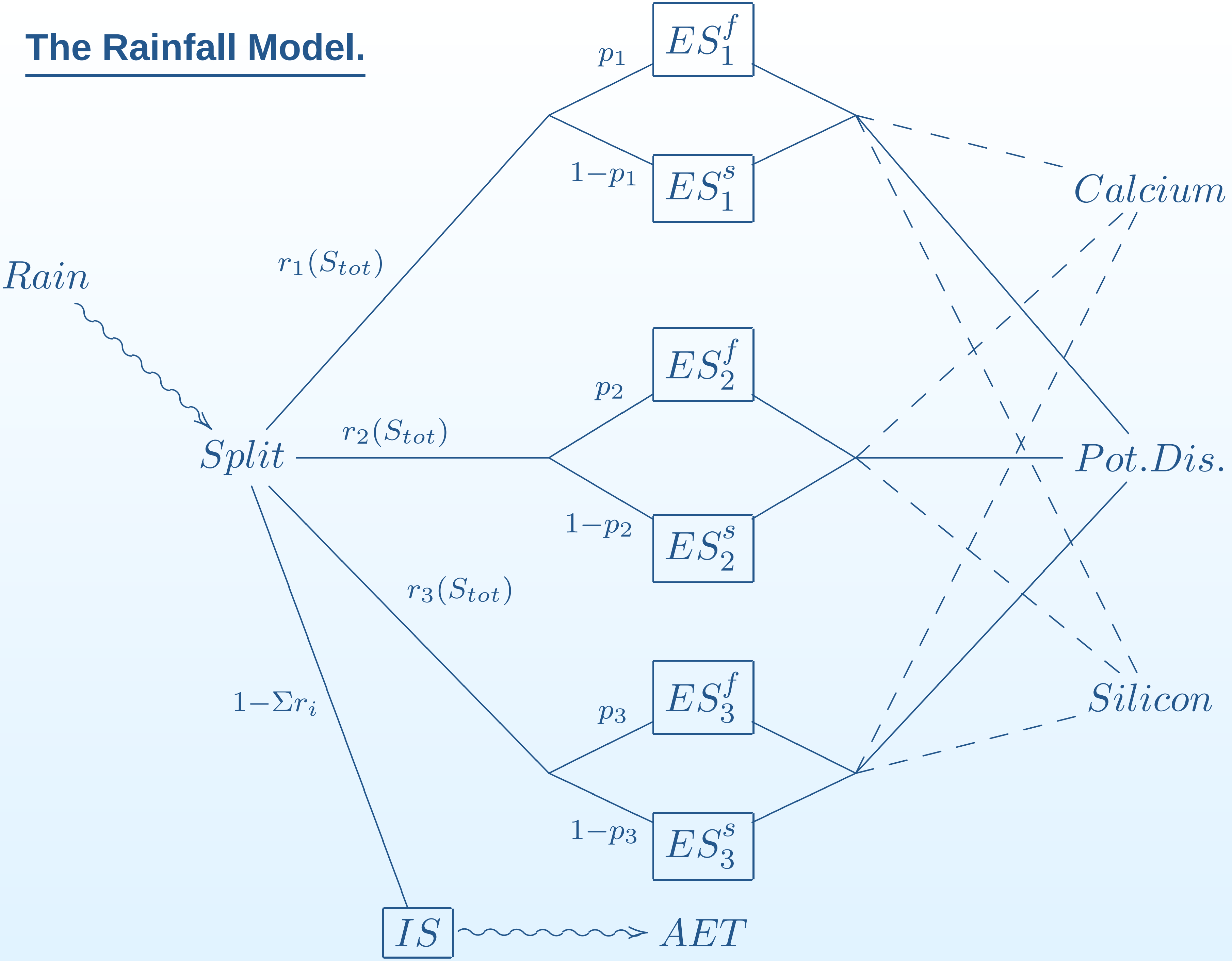
Example of Model output







The Rainfall Model.



Dynamic Emulation

- The model runs over 829 time steps, and produces outputs at each step.
- However, we can consider the model as composed of a 1-step simulator that relates the state of the system at time $t-1$ to the state at time t. This simulator is applied iteratively 839 times.

Dynamic Emulation

- The model runs over 829 time steps, and produces outputs at each step.
- However, we can consider the model as composed of a 1-step simulator that relates the state of the system at time $t - 1$ to the state at time t. This simulator is applied iteratively 839 times.
- Therefore we can emulate the 1-step simulator $f^{(1)}$ as it will be much simpler and this will allow us to track various uncertainties through time (e.g. forcing function uncertainty, model discrepancy).

Dynamic Emulation

- The model runs over 829 time steps, and produces outputs at each step.
- However, we can consider the model as composed of a 1-step simulator that relates the state of the system at time $t-1$ to the state at time t. This simulator is applied iteratively 839 times.
- Therefore we can emulate the 1-step simulator $f^{(1)}$ as it will be much simpler and this will allow us to track various uncertainties through time (e.g. forcing function uncertainty, model discrepancy).
- $f^{(1)}$ is a function that has 26 inputs: the 17 usual inputs to the model x, the forcing functions at time t $(rain(t), AET(t)) = A(t)$ and the 7 State Variables at time t-1: $(ES_i^{f/s}(t-1), S_{tot}(t-1)) = S(t-1)$.
- $f^{(1)}$ has 7 outputs that are the State Variables at time t: $(ES_i^{f/s}(t), S_{tot}(t)) = S(t)$.

Dynamic Emulation and Reification

- Dynamic Emulation of the one step function $f^{(1)}(x, A, S)$ is tricky.

Dynamic Emulation and Reification

- Dynamic Emulation of the one step function $f^{(1)}(x, A, S)$ is tricky.
- Often an emulator for $f^{(1)}(x, A, S)$ that has a high degree of accuracy will still perform poorly over the 839 time step iterations.

Dynamic Emulation and Reification

- Dynamic Emulation of the one step function $f^{(1)}(x, A, S)$ is tricky.
- Often an emulator for $f^{(1)}(x, A, S)$ that has a high degree of accuracy will still perform poorly over the 839 time step iterations.
- The 7 state variable inputs S have such a dominant effect on the outputs of $f^{(1)}(x, A, S)$, it can be hard to pick up the more subtle impact of the other inputs.

Dynamic Emulation and Reification

- Dynamic Emulation of the one step function $f^{(1)}(x, A, S)$ is tricky.
- Often an emulator for $f^{(1)}(x, A, S)$ that has a high degree of accuracy will still perform poorly over the 839 time step iterations.
- The 7 state variable inputs S have such a dominant effect on the outputs of $f^{(1)}(x, A, S)$, it can be hard to pick up the more subtle impact of the other inputs.
- Emulation techniques such as active variables can be misleading.

Dynamic Emulation and Reification

- Dynamic Emulation of the one step function $f^{(1)}(x, A, S)$ is tricky.
- Often an emulator for $f^{(1)}(x, A, S)$ that has a high degree of accuracy will still perform poorly over the 839 time step iterations.
- The 7 state variable inputs S have such a dominant effect on the outputs of $f^{(1)}(x, A, S)$, it can be hard to pick up the more subtle impact of the other inputs.
- Emulation techniques such as active variables can be misleading.
- Accounting for other sources of uncertainty, as required in the Reification process, can still be difficult at this level (although is far easier than Direct Reification).

Modular Emulation and Reification

- Dynamic Emulation requires identification and access to the State variables of the model.

Modular Emulation and Reification

- Dynamic Emulation requires identification and access to the State variables of the model.

- If, in addition, we have access to some or all of the internal variables of the model, we can explore its internal processes and attempt to perform Modular Emulation.

Modular Emulation and Reification

- Dynamic Emulation requires identification and access to the State variables of the model.
- If, in addition, we have access to some or all of the internal variables of the model, we can explore its internal processes and attempt to perform Modular Emulation.
- Can emulate the more subtle parts of the model that may be masked in the Dynamic case.

Modular Emulation and Reification

- Dynamic Emulation requires identification and access to the State variables of the model.
- If, in addition, we have access to some or all of the internal variables of the model, we can explore its internal processes and attempt to perform Modular Emulation.
- Can emulate the more subtle parts of the model that may be masked in the Dynamic case.
- Better for our purpose of linking the model to reality: simpler to incorporate other sources of uncertainty.

Modular Emulation and Reification

- Dynamic Emulation requires identification and access to the State variables of the model.
- If, in addition, we have access to some or all of the internal variables of the model, we can explore its internal processes and attempt to perform Modular Emulation.
- Can emulate the more subtle parts of the model that may be masked in the Dynamic case.
- Better for our purpose of linking the model to reality: simpler to incorporate other sources of uncertainty.
- Easier for expert to consider improvements to an individual process, as opposed to the whole model.

Modular Emulation and Reification

- Dynamic Emulation requires identification and access to the State variables of the model.
- If, in addition, we have access to some or all of the internal variables of the model, we can explore its internal processes and attempt to perform Modular Emulation.
- Can emulate the more subtle parts of the model that may be masked in the Dynamic case.
- Better for our purpose of linking the model to reality: simpler to incorporate other sources of uncertainty.
- Easier for expert to consider improvements to an individual process, as opposed to the whole model.
- Downside: it is difficult to link the modular emulators together (but we already have this problem in the Dynamic Case).

Modular Emulation and Reification

- To incorporate other sources of uncertainty and to perform the Reification technique, we require a satisfactory Modular Emulator.

Modular Emulation and Reification

- To incorporate other sources of uncertainty and to perform the Reification technique, we require a satisfactory Modular Emulator.
- Note that for one run of the full model, we can obtain 839 runs of the one-step function and of each of the 3 modular functions, for the same cost.

Modular Emulation and Reification

- To incorporate other sources of uncertainty and to perform the Reification technique, we require a satisfactory Modular Emulator.
- Note that for one run of the full model, we can obtain 839 runs of the one-step function and of each of the 3 modular functions, for the same cost.
- We identified active variables, and constructed fast emulators composed of linear models with 2nd order polynomials in the active variables.

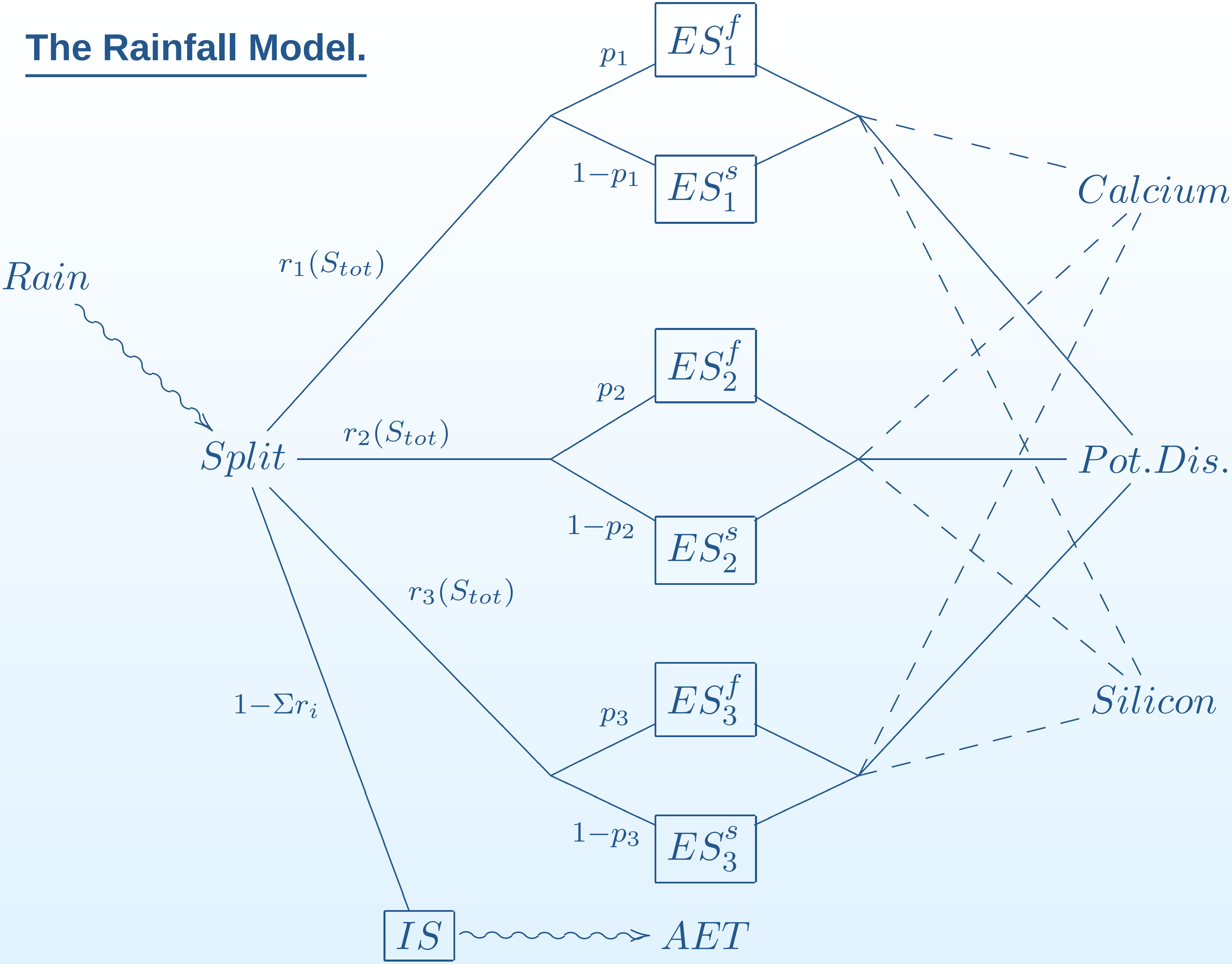
Modular Emulation and Reification

- To incorporate other sources of uncertainty and to perform the Reification technique, we require a satisfactory Modular Emulator.
- Note that for one run of the full model, we can obtain 839 runs of the one-step function and of each of the 3 modular functions, for the same cost.
- We identified active variables, and constructed fast emulators composed of linear models with 2nd order polynomials in the active variables.
- Combined modular emulators using structure of the model, with linearised approximations to calculate propagation of uncertainties where necessary.

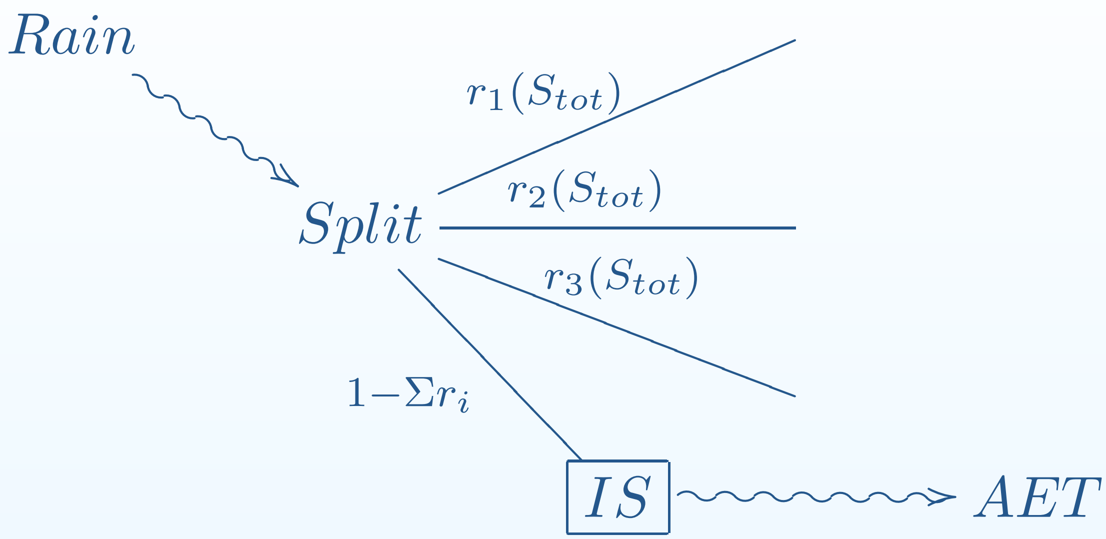
Modular Emulation and Reification

- To incorporate other sources of uncertainty and to perform the Reification technique, we require a satisfactory Modular Emulator.
- Note that for one run of the full model, we can obtain 839 runs of the one-step function and of each of the 3 modular functions, for the same cost.
- We identified active variables, and constructed fast emulators composed of linear models with 2nd order polynomials in the active variables.
- Combined modular emulators using structure of the model, with linearised approximations to calculate propagation of uncertainties where necessary.
- Intend to improve emulators at later stage.

The Rainfall Model.

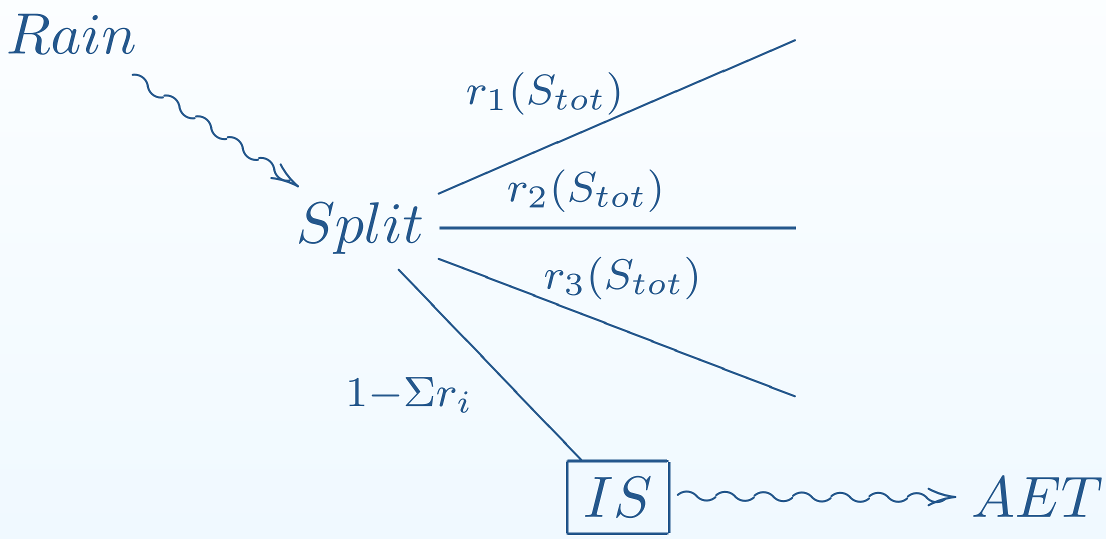


Module 1



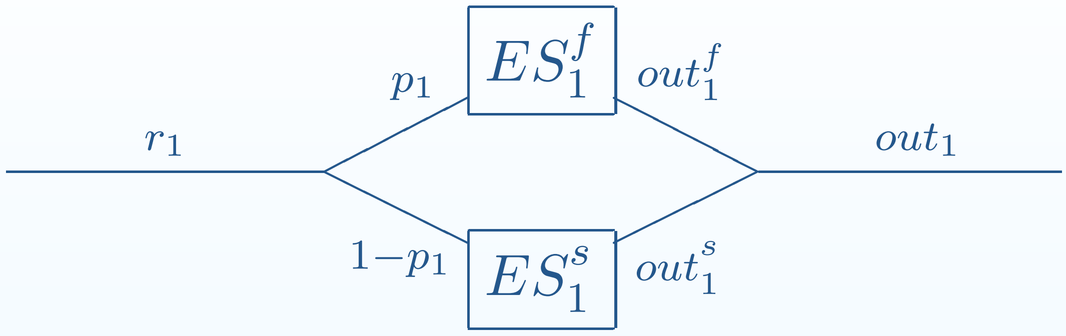
- Emulated the split function outputs r_i to reasonably high degree of accuracy.
- Find that the r_i depend on only $k_i, a_i, b_i, S_{tot}, rain$.

Module 1



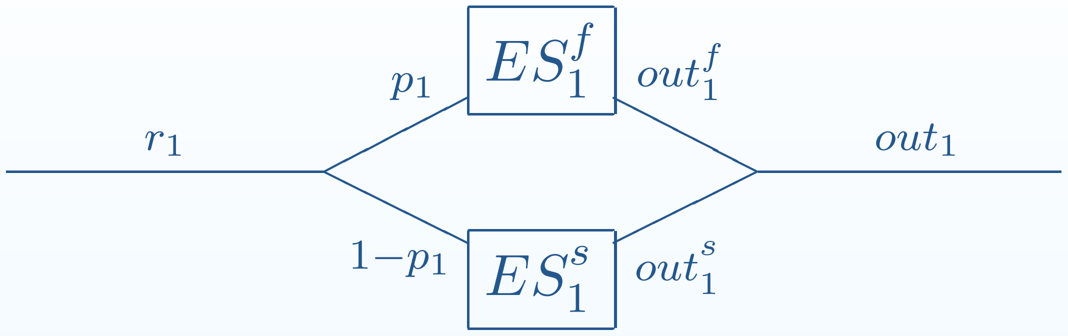
- Emulated the split function outputs r_i to reasonably high degree of accuracy.
- Find that the r_i depend on only $k_i, a_i, b_i, S_{tot}, rain$.
- Allowed incorporation of the uncertain forcing function rainfall.
- Will attempt to do internal reification on the r_i functions directly.

Module 2



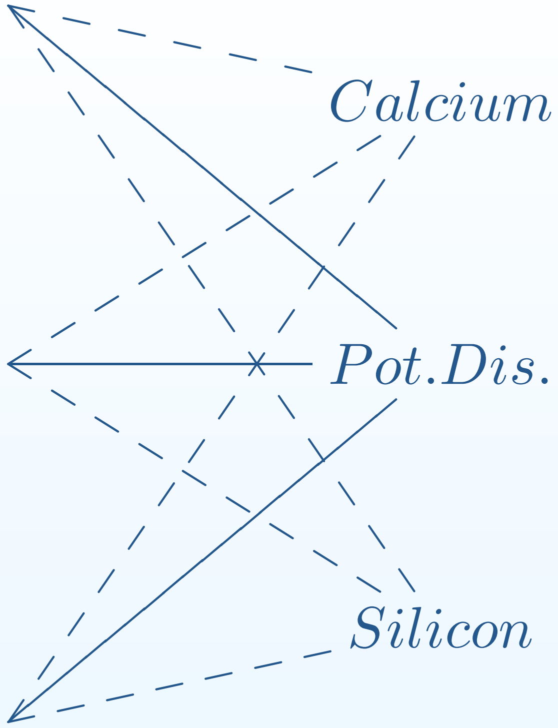
- Similarly we want to emulate and reify the above single ’compartment’
- We can choose how far ‘into the box’ we want to go.

Module 2



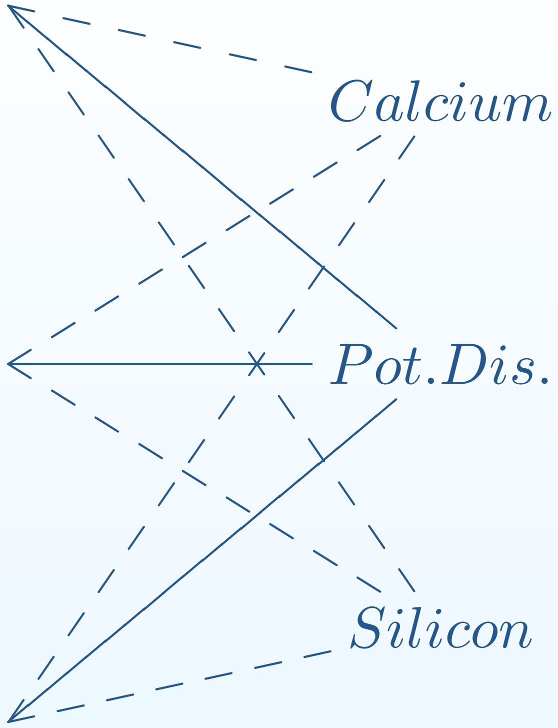
- Similarly we want to emulate and reify the above single ’compartment’
- We can choose how far ‘into the box’ we want to go.
- e.g. we find that $out_1^f(t)$ is a function that only depends on two inputs: $ES_1^f(t-1)$ and cf_1, and is trivial to emulate.
- Again can Reify this (or parts of this) compartment process.

Module 3



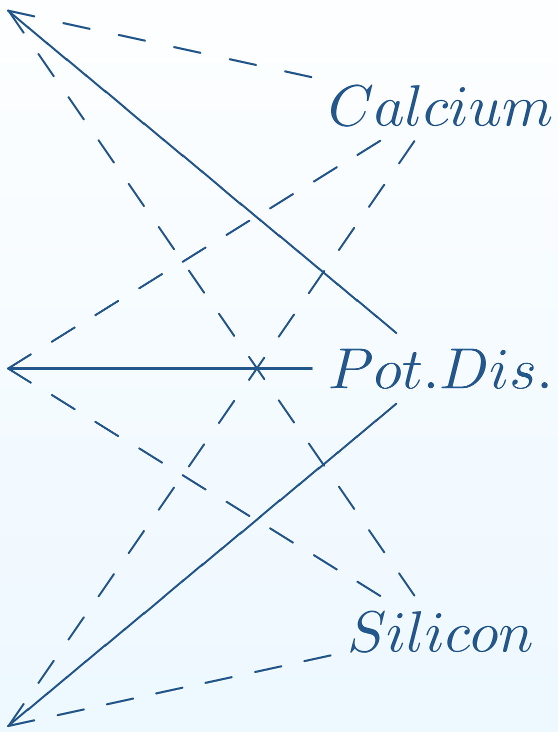
- Final outputs Calcium, Pot.Dis. and Silicon straightforward to emulate.

Module 3



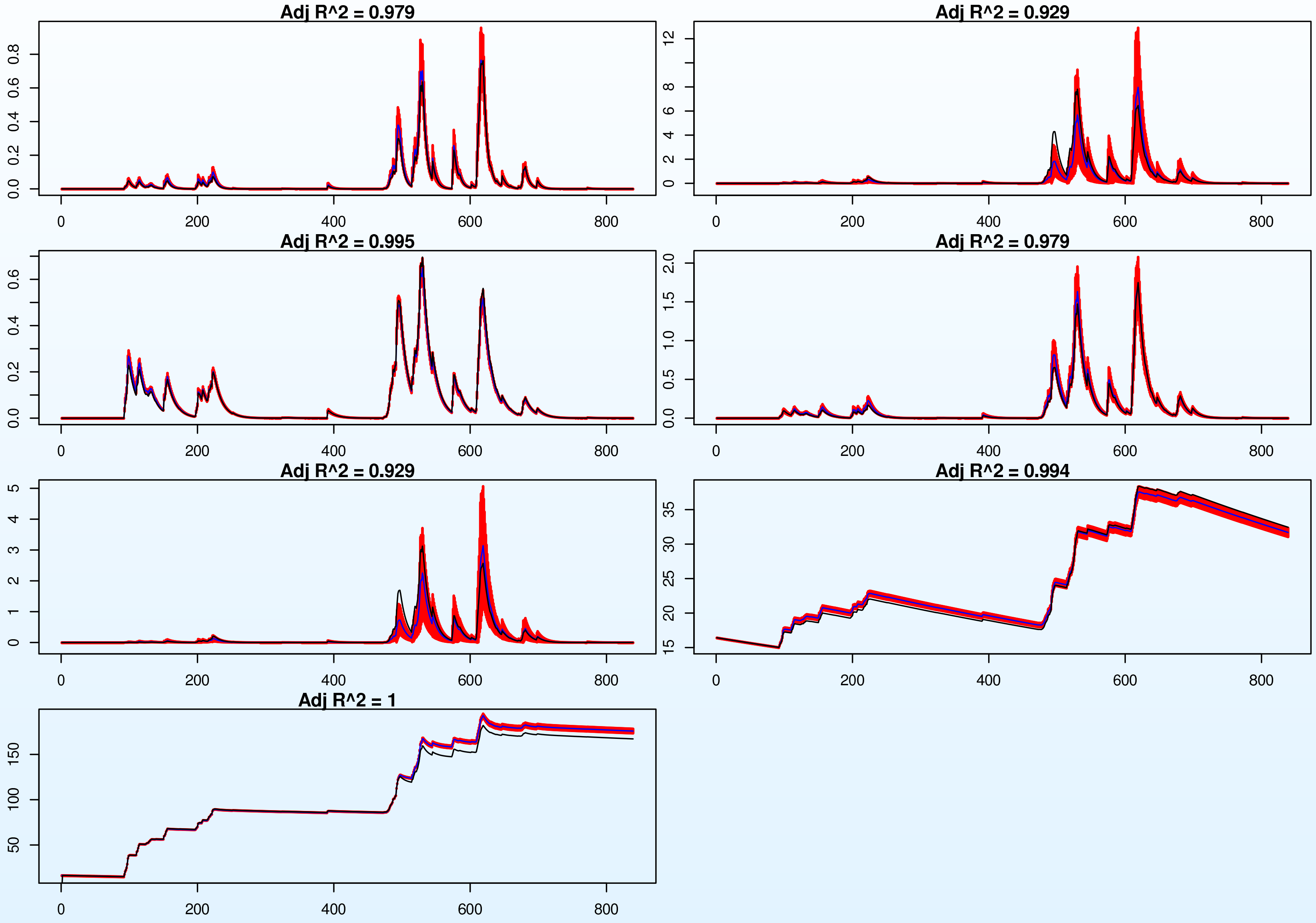
- Final outputs Calcium, Pot.Dis. and Silicon straightforward to emulate.
- Contains assumption of fixed Tracers that lead to Ca and Si concentration in rivers.

Module 3

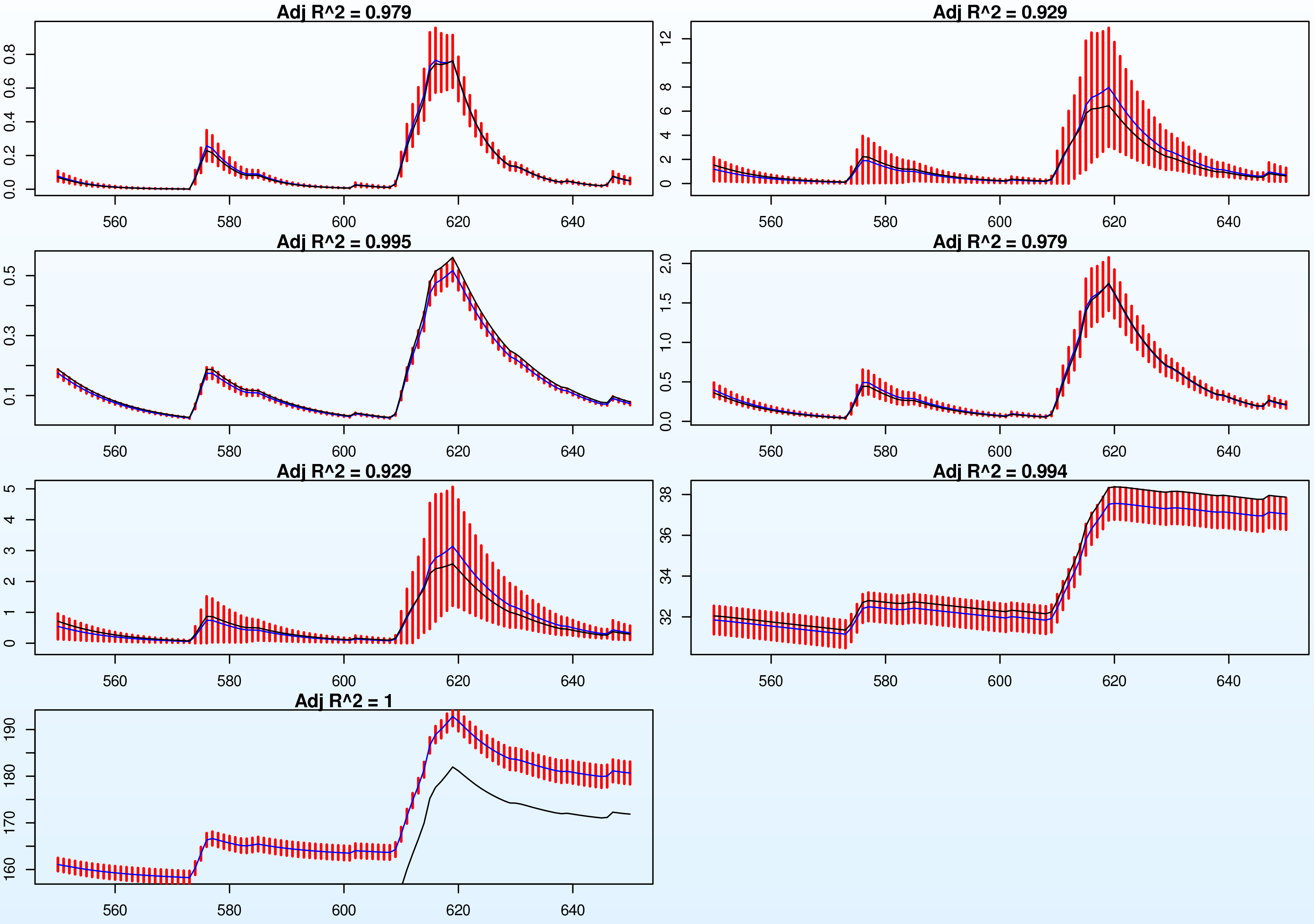


- Final outputs Calcium, Pot.Dis. and Silicon straightforward to emulate.
- Contains assumption of fixed Tracers that lead to Ca and Si concentration in rivers.
- Modular Emulation has allowed incorporation of uncertainty due to Tracers: first step in a Reification of this function.

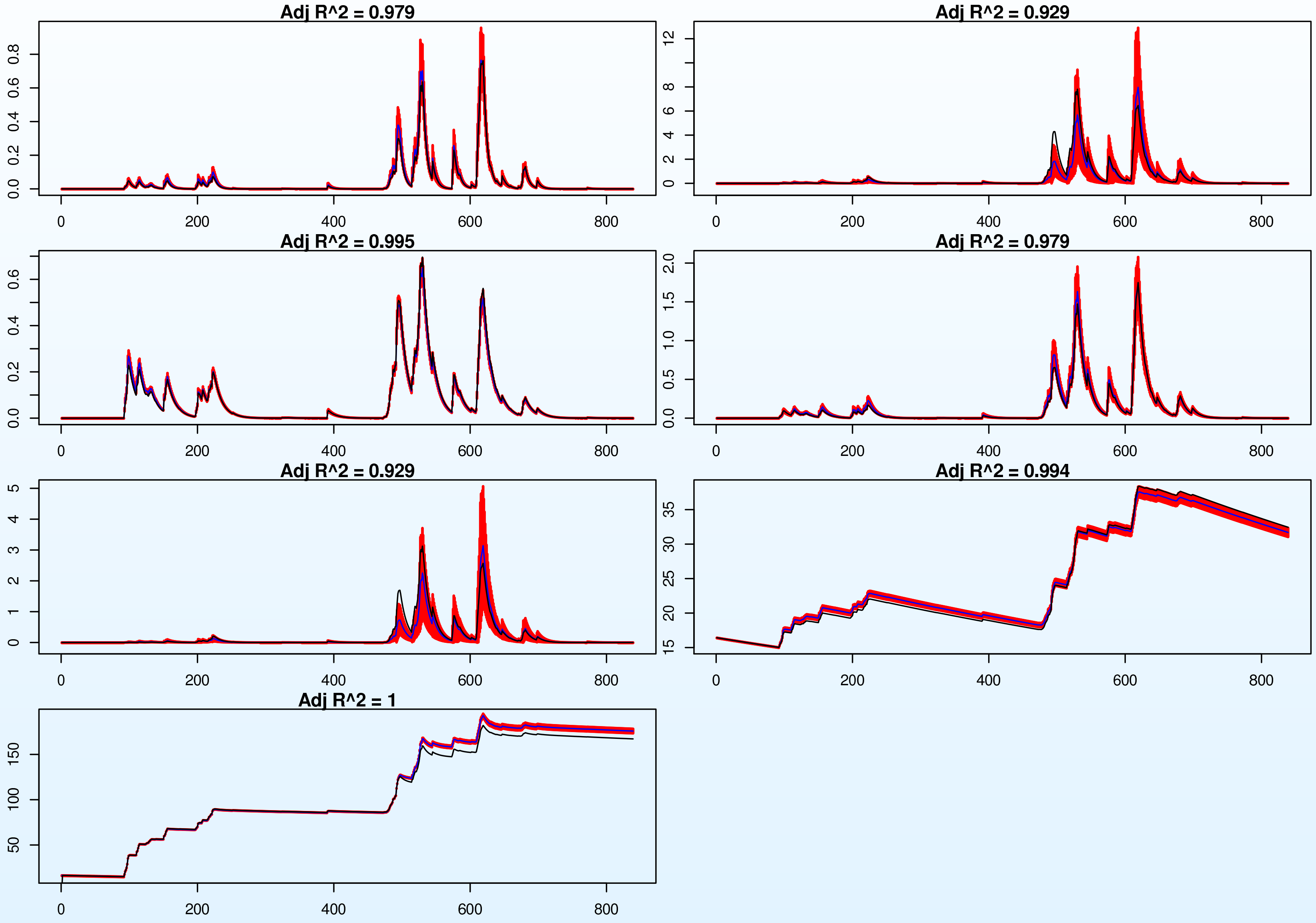
The Modular Emulator: State Variables



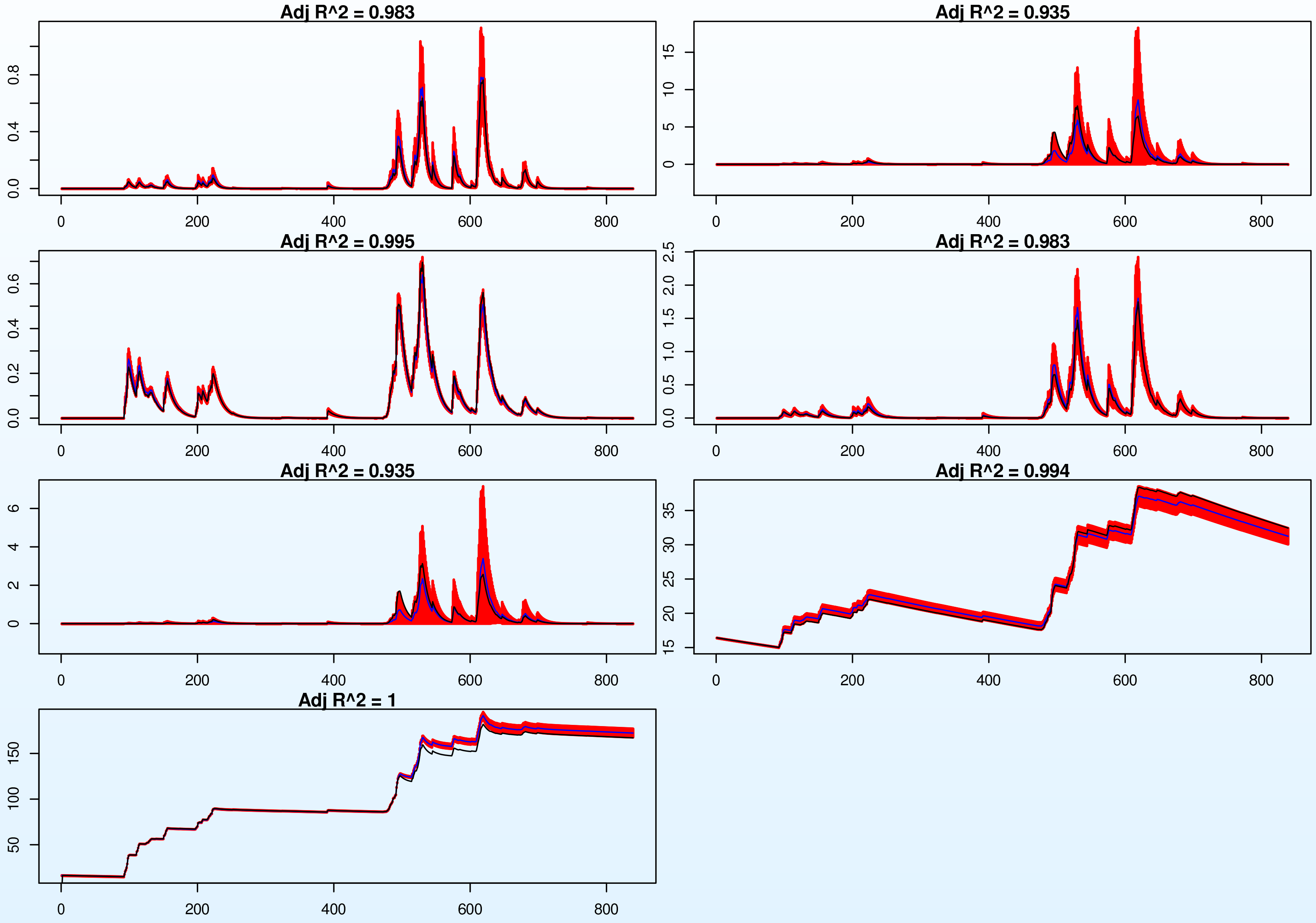
The Modular Emulator: State Variables



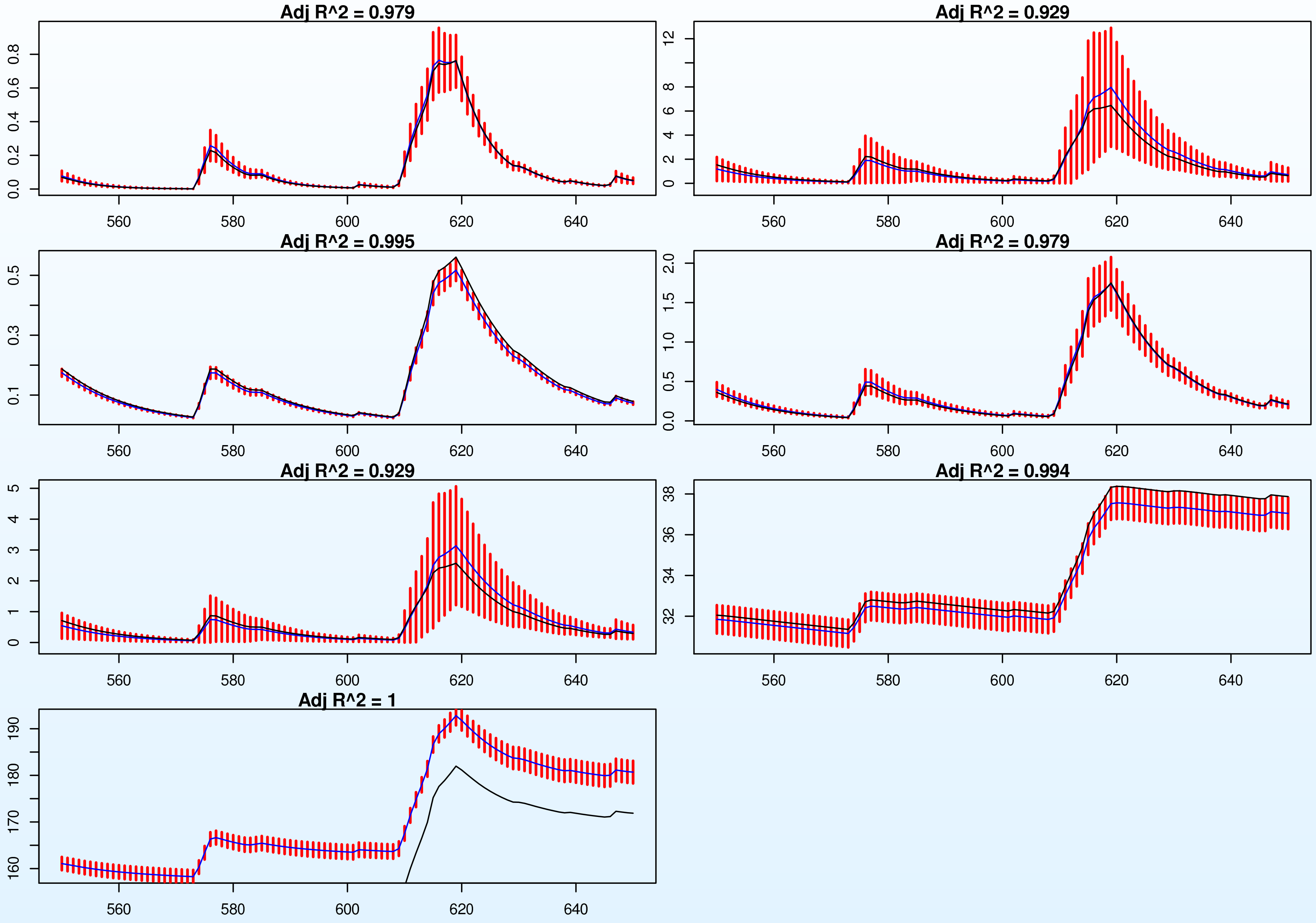
The Modular Emulator: State Variables



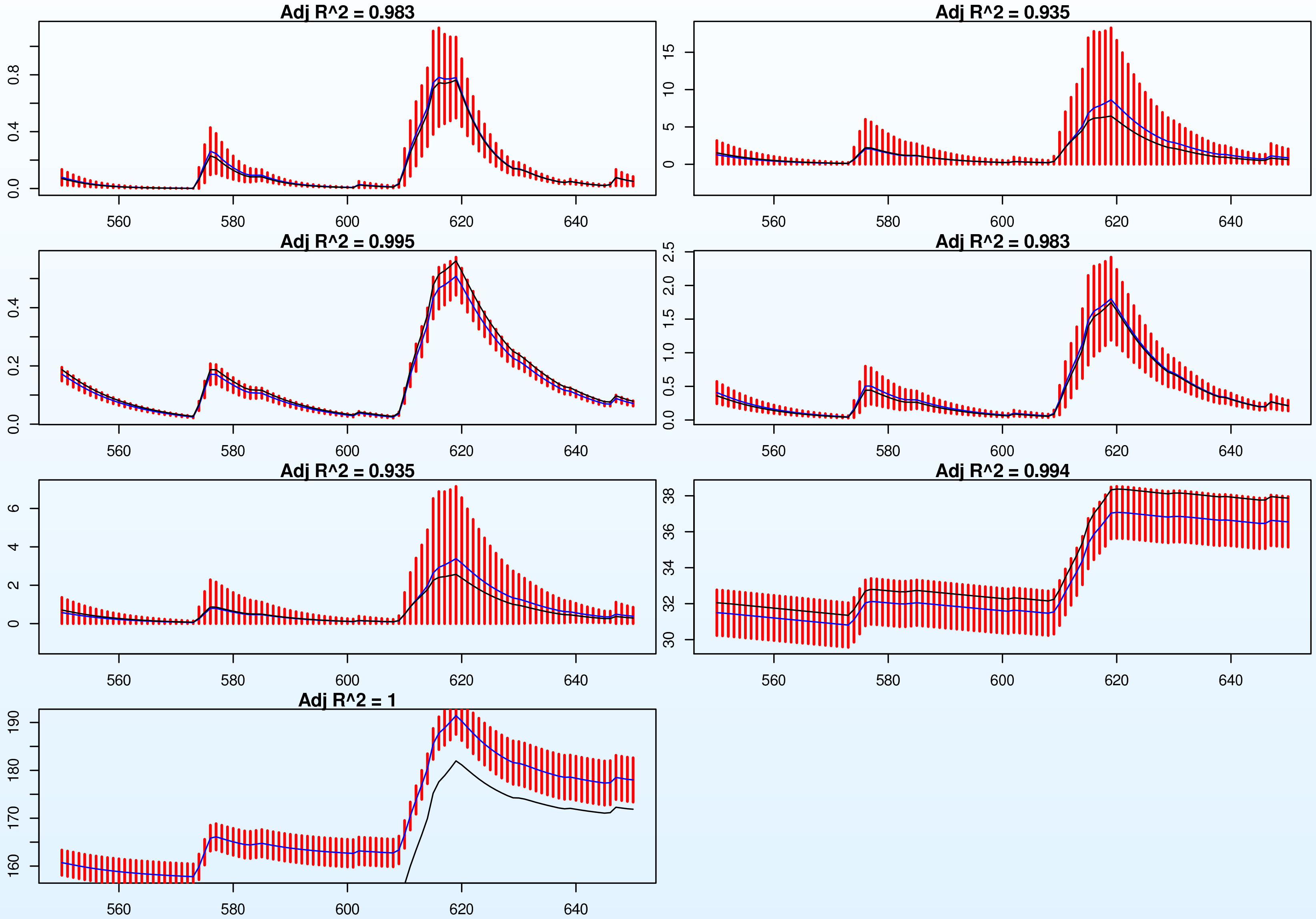
The Modular Emulator: Model Discrepancy



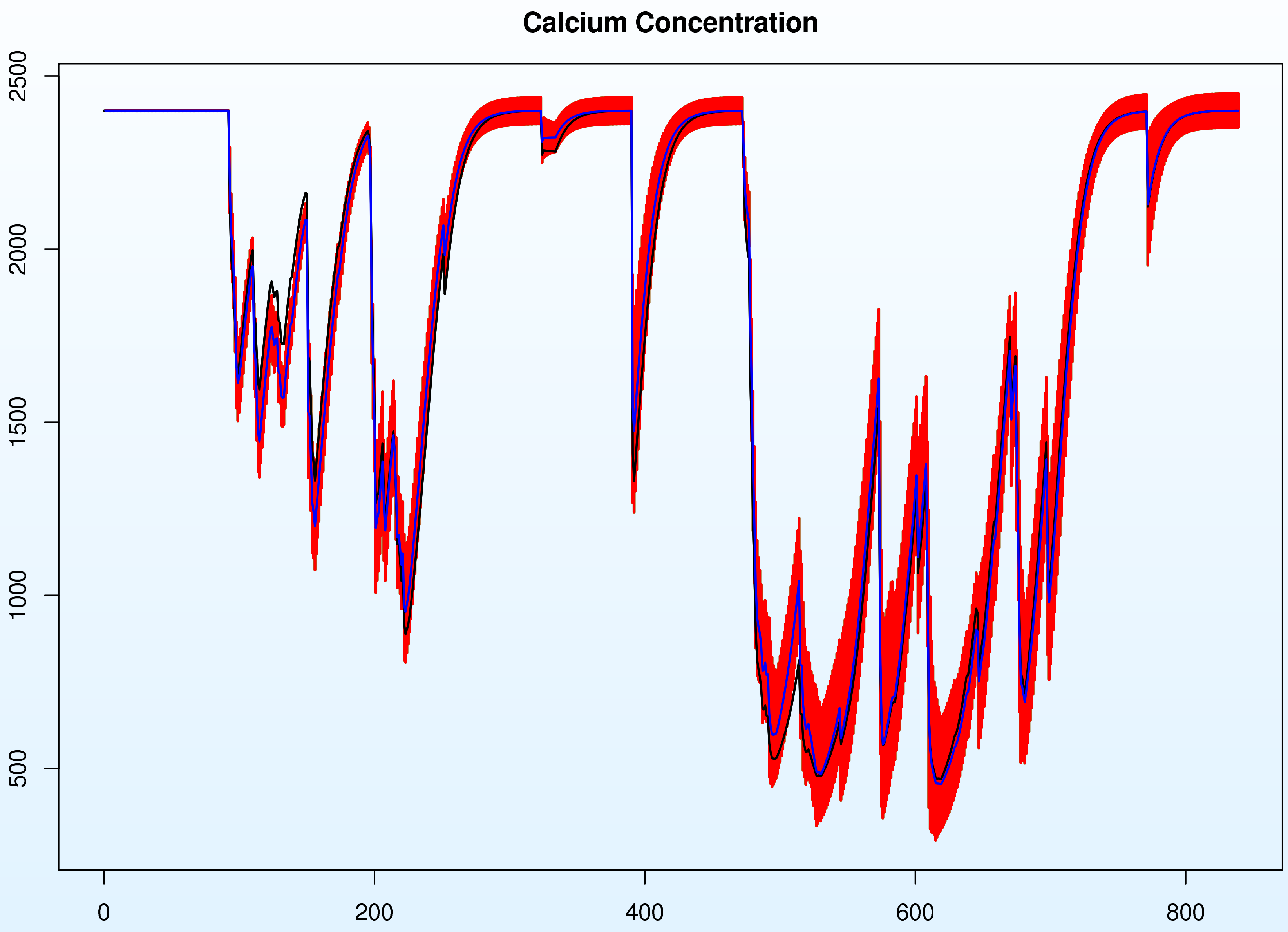
The Modular Emulator: State Variables



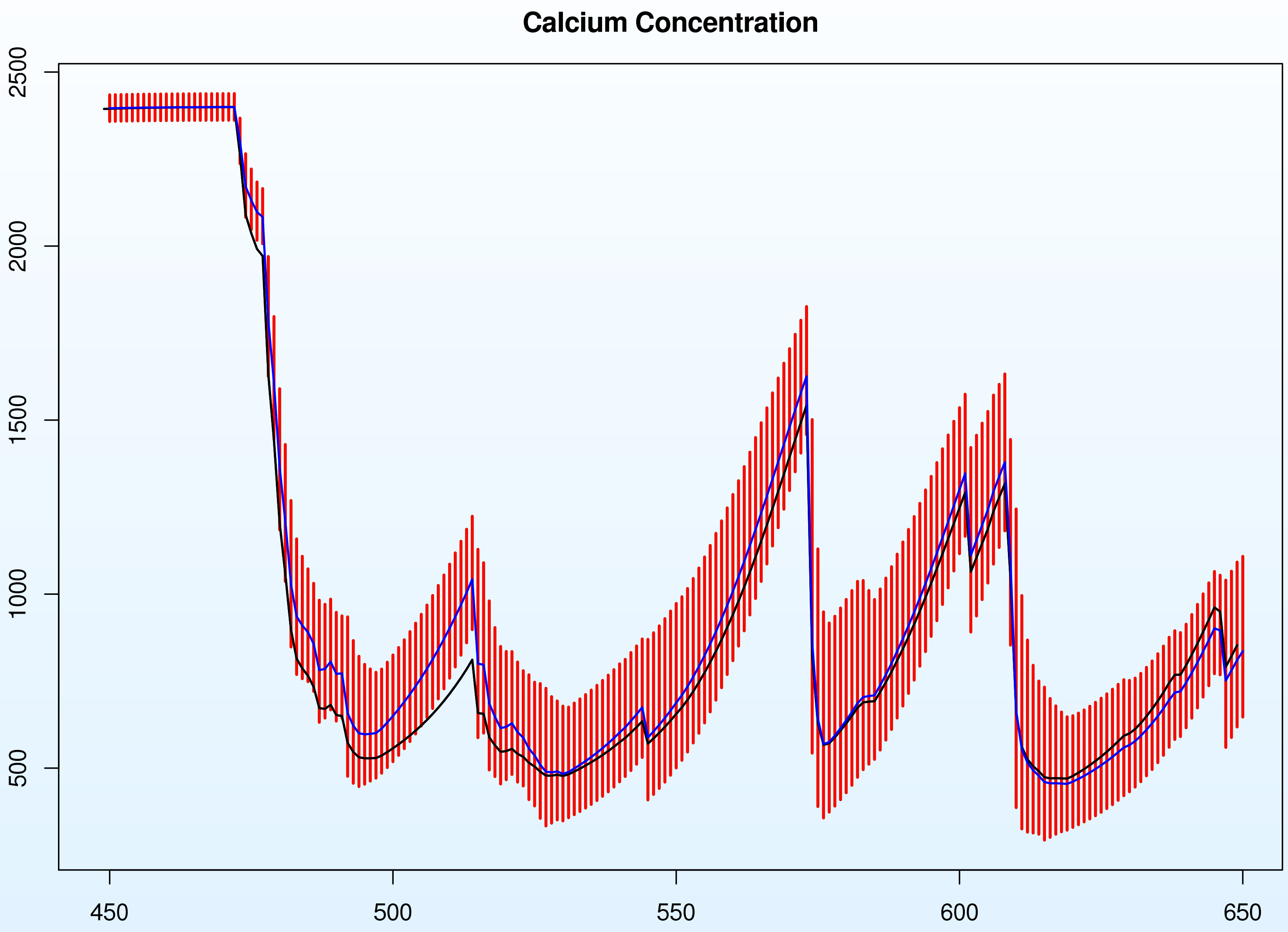
The Modular Emulator: Model Discrepancy



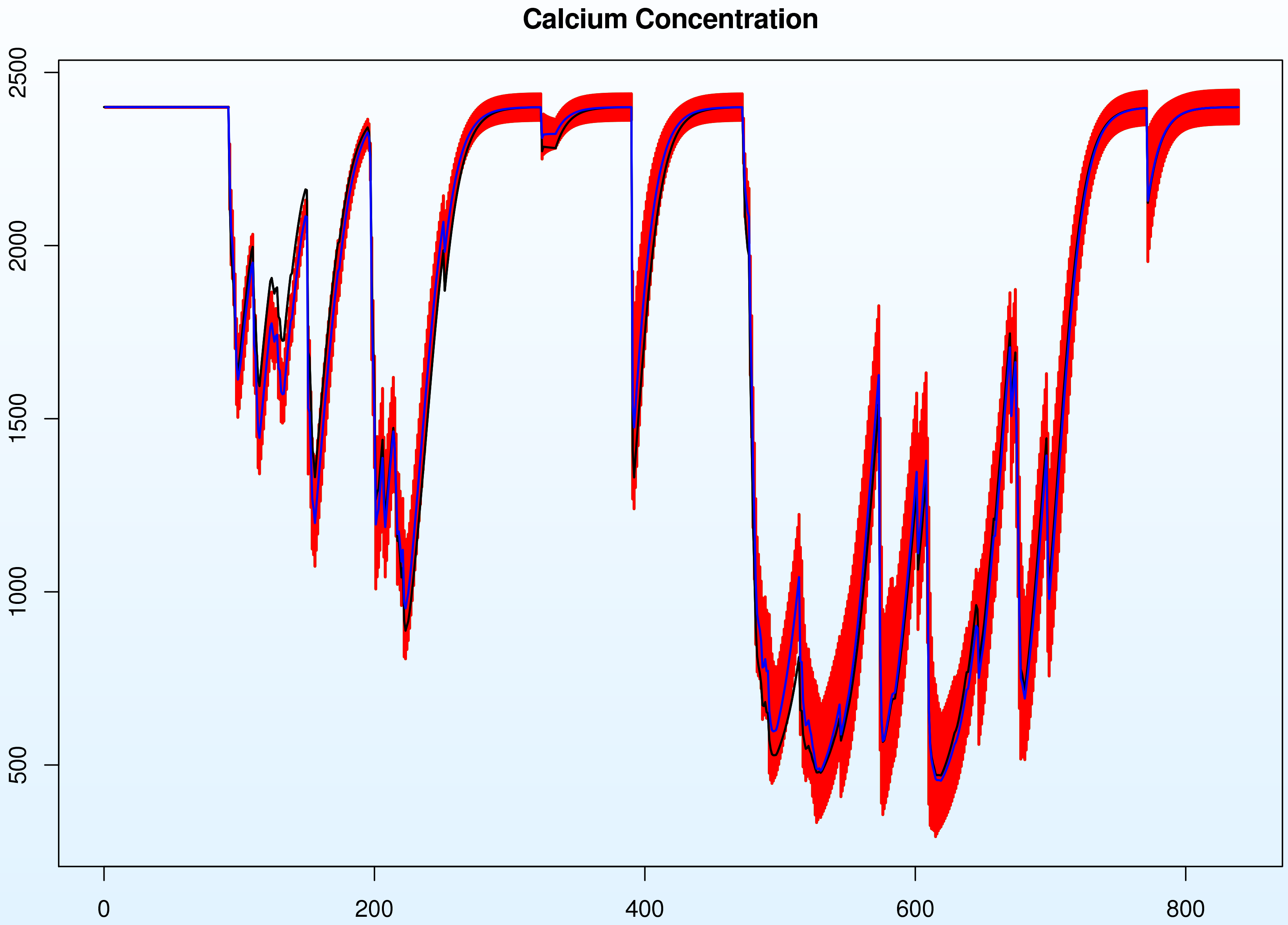
The Modular Emulator: Calcium Output



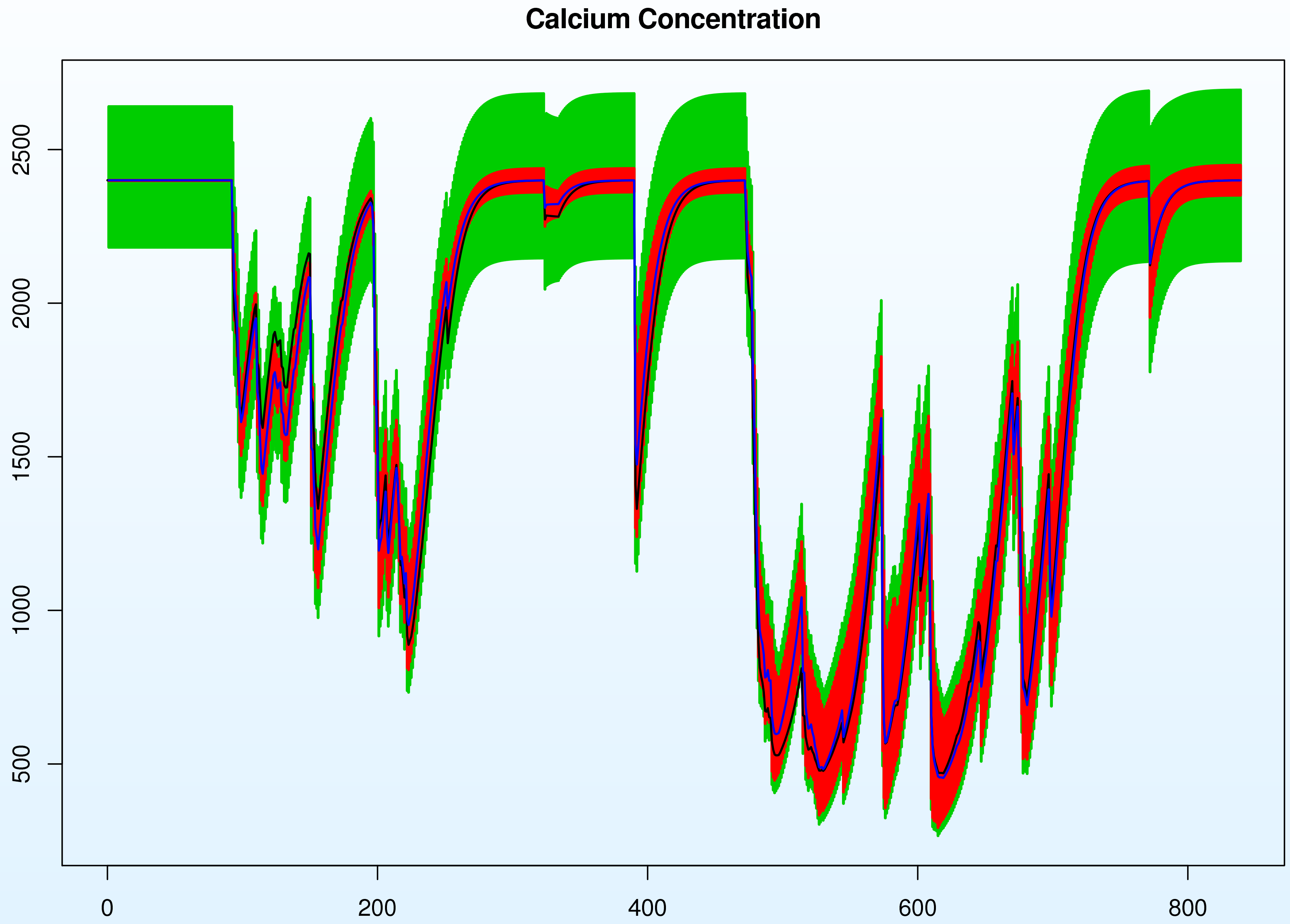
The Modular Emulator: Calcium Output



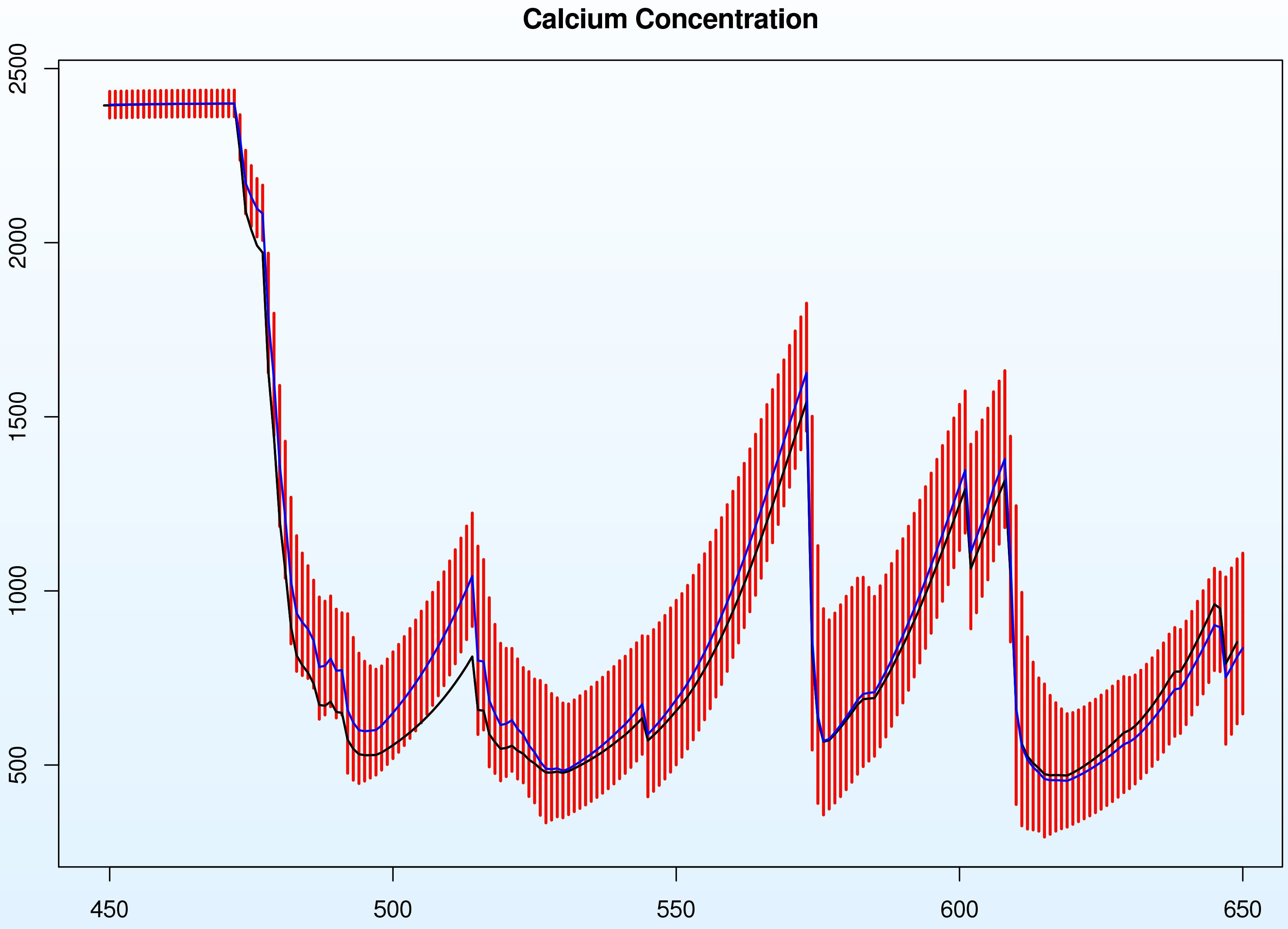
The Modular Emulator: Tracer Uncertainty



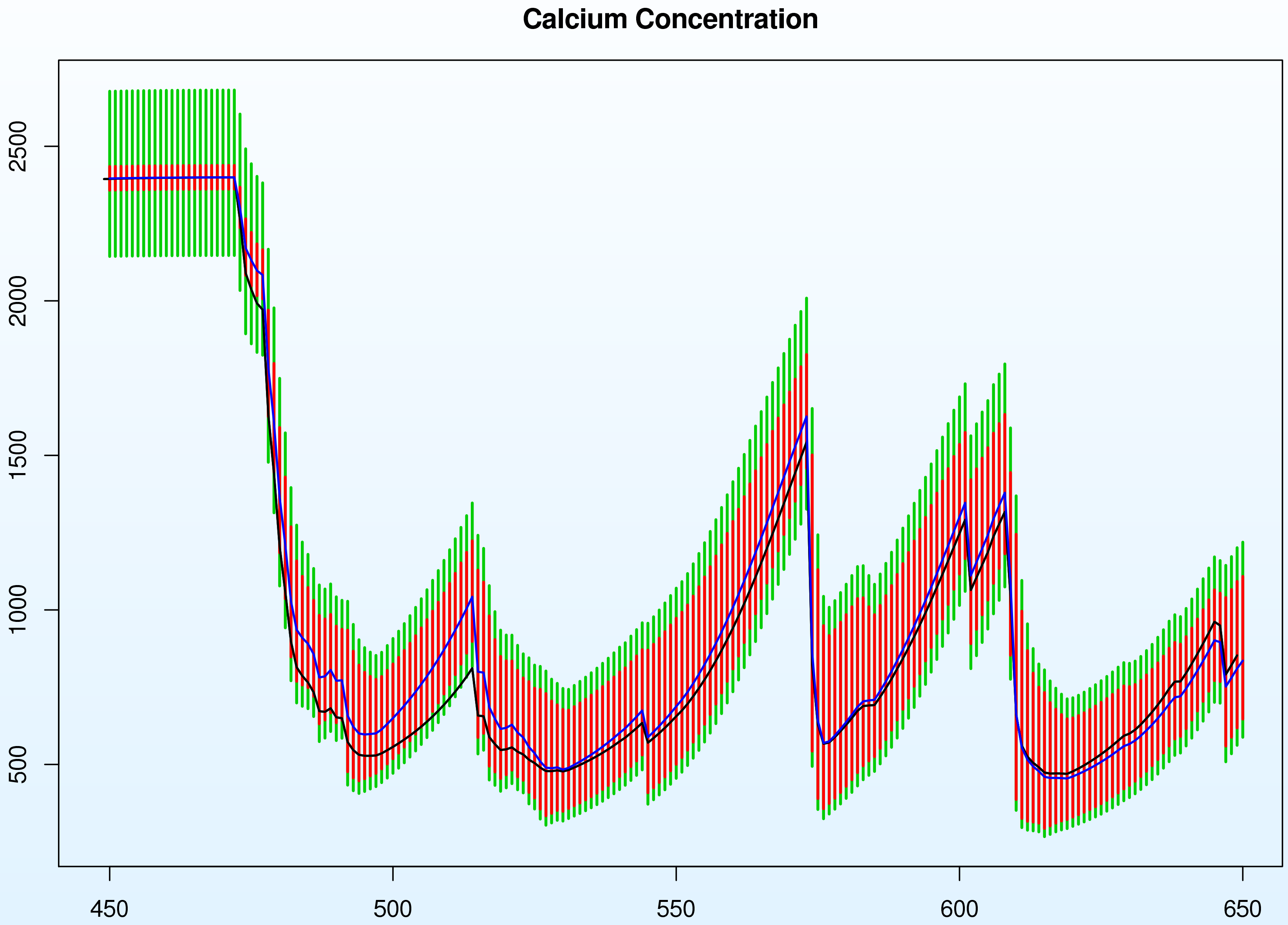
The Modular Emulator: Tracer Uncertainty



The Modular Emulator: Tracer Uncertainty



The Modular Emulator: Tracer Uncertainty



Conclusions

- We have developed fast Modular Emulators of sufficient accuracy that are required for inclusion of detailed uncertainties.

Conclusions

- We have developed fast Modular Emulators of sufficient accuracy that are required for inclusion of detailed uncertainties.
- Have incorporated uncertainty due to the Tracer concentrations, simple internal model discrepancy, and the rainfall forcing function.

Conclusions

- We have developed fast Modular Emulators of sufficient accuracy that are required for inclusion of detailed uncertainties.
- Have incorporated uncertainty due to the Tracer concentrations, simple internal model discrepancy, and the rainfall forcing function.
- Lots more to do: Full Structural Reification on the Dynamic or Modular level (e.g. 4th compartment, output functions, split function...).

Conclusions

- We have developed fast Modular Emulators of sufficient accuracy that are required for inclusion of detailed uncertainties.
- Have incorporated uncertainty due to the Tracer concentrations, simple internal model discrepancy, and the rainfall forcing function.
- Lots more to do: Full Structural Reification on the Dynamic or Modular level (e.g. 4th compartment, output functions, split function...).
- Lots more to say: Checking assumptions via simulation, comparisons of Direct Emulation with Dynamic and Modular Emulation, assessment of relative impact of different uncertainties, upgrading constant inputs to time dependent processes, External Reification...

References

M. Goldstein and J.C.Rougier. Probabilistic formulations for transferring inferences from mathematical models to physical systems (2005) SIAM journal on scientific computing, 26, 467-487.

M. Goldstein and J.C.Rougier. Reified Bayesian modelling and inference for physical systems (2006) In revision for JSPI.