**Protocol of reconstructing model structure by data assimilation**

**Demo code repository:** [**https://github.com/phxtao/TRENDY\_CLM5**](https://github.com/phxtao/TRENDY_CLM5)

1. **Data input** 
   1. [demo code/src\_da/da\_reconstruct.m #L79]
   2. model dependent, any relevant variables that drive the specific transient model simulation.
   3. e.g., input variables that drive CLM5 simulation:

|  |  |
| --- | --- |
| Variable name in CLM5 | Full description |
| nbedrock | Soil layer number that reaches the bedrock |
| ALTMAX | Maximum active layer depth of current year |
| ALTMAX\_LASTYEAR | Maximum active layer depth of last year |
| CELLSAND | Sand content |
| NPP | Net primary productivity |
| SOILPSI | Soil water potential |
| TSOI | Soil temperature |
| O\_SCALAR | Oxygen scalar for decomposition |
| FPI\_vr | Nitrogen scalar for decomposition |

1. **“Observations”**
   1. [demo code/src\_da/da\_reconstruct.m #L82]
   2. Original model outputs about **carbon pool and flux** from either TRENDY or CMIP6.
   3. e.g., CLM5: total SOC pool, total heterotrophic respiration
2. **Forward model simulation function** 
   1. [demo code/src\_clm\_cen/fun\_forward\_simu.m]
   2. Capacity of simulating a predefined period of land carbon cycle dynamics
   3. Generating target output (to be compared with observations)
3. **MCMC algorithm** 
   1. [demo code/src\_da/da\_reconstruct.m #L138-404]
   2. **Independent series of MCMC:** at least three series of MCMC are required (for assessing the convergence)
   3. **Cost function** 
      1. Define cost function at [demo code/ src\_clm\_cen/cost\_fun.m]
      2. Call cost function at [demo code/src\_da/da\_reconstruct.m #L136, #L216, #L339]
      3. Single constraint (*j*): , where **m** represents the model used in MCMC, *mod*i denotes the modeled results, *obs*i represents the observations; σi is the standard deviation of the observations; and k is the number of observations. σi is a value from our empirical estimates and can have influence in the final acceptance ratio. Default is 0.3\**obs*, but can be adjusted depending on different models.
      4. Multiple constraints:
         * , where is a coefficient representing the weighting of constraint *j* in calculating cost function. Default = 1 but can be adjusted depending on different models.
         * Another option you can consider:
   4. **Test run** 
      1. [demo code/src\_da/da\_reconstruct.m #L156-263]
      2. Prior distribution assumption: uniform distributions. Prior ranges of the uniform distribution are from empirical estimates [demo code/ src\_clm\_cen/fun\_forward\_simu.m #L106 - #L156]
      3. Proposal [demo code/src\_da/da\_reconstruct.m #172]: , where *θ max*and *θ min*are the upper and lower limits of parameter values (Table S1). *r* is a uniformly distributed random variable over [-0.5, 0.5]. *D* is a coefficient controlling the step size of the newly proposed parameter value. Default value of *D* = 5 and can be adjusted depending on different models*.*
      4. Iteration number [demo code/src\_da/da\_reconstruct.m #L116]: recommend 10,000 to 30,000 iterations (at least ~200 sets of parameters accepted after test run)
   5. **Formal run**
      1. [demo code/src\_da/da\_reconstruct.m #L246-404]
      2. Prior distribution assumption: multivariate normal distribution [demo code/src\_da/da\_reconstruct.m #L270 #L370]
      3. Proposal [demo code/src\_da/da\_reconstruct.m #L293]: , where we calculate the covariance matrix of parameters by the results of the test run at the starting stage of the formal run (i.e., before the iteration). The is a multiplier that only depends on the number of investigated parameters for the best efficiency of the MCMC simulation. After some iterations of the formal run, we continuously update the covariance information from the formal run results. The point is an empirical value we may set based on experience, depending on how fast we think the formal run can fully utilize the prior information in the test run. We recommend set default and as a start. Both of them can be adjusted in order to obtain better data assimilation results
      4. Iteration number [demo code/src\_da/da\_reconstruct.m #L117]: 30,000 to 100,000
4. **Assessment of data assimilation performance**
   1. **Acceptance ratio:** 
      1. Only use the results of formal run in calculation
      2. Criteria: control the AR between 10% - 50%
   2. **G-R statistic (indicator of MCMC convergence)** [demo code/src\_da/da\_reconstruct.m #L407 - #L426]
      1. If parameter chains have reached convergence, the within-run variation should be roughly equal to the between-run variation. The within-run and between-run variation are expressed as: where *i* denotes parameters investigated in this study; *K* is the number of parallel runs; *N* is the length of each run; represents the nth accepted value of parameter *i* in the kth parallel run after the burn-in period; and the length of burn-in period was set to be half of the accepted parameter chain.
      2. The G-R statistics is defined as:
      3. Criteria: Once the convergence is reached, *GRi* should approximately approach 1. We recommend calculate the mean value of for different parameters () and control .
   3. **Coefficient of efficiency** [demo code/src\_da/da\_reconstruct.m #L428 - #L450]
      1. . The *E* is equivalent to the coefficient of determination in a linear regression when the linear model is set as y = x (1:1 line).
      2. Criteria: The larger E, the better the MCMC performs
5. **Tuning hyper-parameters in MCMC for better performance**
   1. ***D* in test run:** 
      1. [demo code/src\_da/da\_reconstruct.m #L158]
      2. A small *D* indicates larger step size 🡪 decrease acceptance ratio
      3. A large *D* indicates smaller step size 🡪 increase acceptance ratio
   2. **Covariance scaling factor () in formal run**
      1. [demo code/src\_da/da\_reconstruct.m #L267-#L268]
      2. A small shrinks the covariance matrix 🡪 increase acceptance ratio
      3. A large amplify the covariance matrix 🡪 decrease acceptance ratio
   3. **Weighting in cost function**
      1. Weighting value should be adjusted by comprehensive assessment of acceptance ratio (efficiency of MCMC) and coefficient of efficiency (how well the model represents observations)
      2. When acceptance ratio is low, set smaller weighting value may help increase acceptance, and vice versa
      3. When coefficient of efficiency of certain types of observations is low, increasing their corresponding weighting value may help the model focus more on that observation in MCMC.