Parameter and initial state estimation (4D-Var)

The ODE system and observation model is given as

$$\frac{dx}{dt} = M(t,\ x, p) \xrightarrow{discretization} x^{(t)} = x^{(t-1)} + \Delta t M(t\Delta t,\ x^{(t)}, p)$$

$$y \sim Normal(H(x), R),$$

where t, x, y, and p denotes time, state variables, observables, and model parameters, respectively. M, H, and R represents for ODE system, observation operator, and covariance matrix for observation, respectively. Estimation problem of parameters and initial state variables can be established as a cost minimization problem subject to the discretized ODE system.

$$\text{Cost function} \quad J\left(x^{(0)}, p\right) = \tilde{J}\left(x^{(0:T)}, p\right) = \sum_{t=0}^{T} \frac{\left(H\left(x^{(t)}\right) - y\right)^{T} R^{-1}\left(H\left(x^{(t)}\right) - y\right)}{2} + L(p).$$

$$\text{Minimize}_{x^{(0)}, p}\ \tilde{J}\left(x^{(0:T)}, p\right) \ \text{s.t.}\ \ x^{(t)} = x^{(t-1)} + \Delta t M(t\Delta t,\ x^{(t)}, p).$$

The cost function is derived from a negative logarithm of the likelihood function; $\mathrm{Prob}\left(y \middle| x^{(0)}, p\right) \propto \exp\left\{-J\left(x^{(0)}, p\right)\right\}$. The minimization problem can be solved as follows using Lagrange multiplier $\lambda^{(t)}$, $\nu^{(t)}$ and dynamic programming.

$$\begin{cases} \lambda^{(T+1)} = 0, \nu^{(T+1)} = 0 \\ \lambda^{(t)} = \left(I - \Delta t \dfrac{\partial M}{\partial x^{(t)}}\right)^{-T} \left(\lambda^{(t+1)} + \dfrac{\partial \tilde{J}}{\partial x^{(t)}}\right) \quad (t = T, \cdots, 1) \\ \nu^{(t)} = \nu^{(t+1)} + \Delta t \left(\dfrac{\partial M}{\partial p}\right)^{T} \lambda^{(t)} \quad (t = T, \cdots, 1) \end{cases}$$

The gradient of the cost function is calculated as follows,

$$\begin{cases} \dfrac{\partial J}{\partial x^{(0)}} = \lambda^{(1)} + \dfrac{\partial \tilde{J}}{\partial x^{(0)}} \\ \dfrac{\partial J}{\partial p} = \nu^{(1)} + \dfrac{\partial L}{\partial p}. \end{cases}$$

Thus, $(x^{(0)}, p)$, the pair of initial state variables and parameters, can be optimized through gradient methods such as limited-memory Broyden–Fletcher–Goldfarb–Shanno (LBFGS) method (J. Nocedal, 1980, *Math. Comput.*).

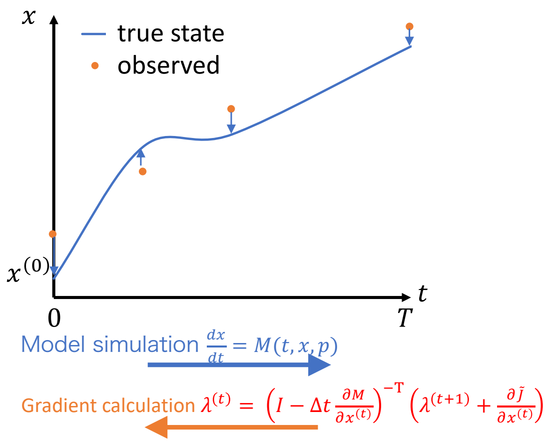

Figure 1. 4D-Var data assimilation.

Confidence interval and covariance calculation

Method to obtain confidence intervals of the estimated parameters and initial state variables. Through variational calculation, Hessian-vector product of the cost function can be obtained solving the following dynamic programming equations,

$$\delta x^{(t)} = \left(I - \Delta t \frac{\partial M}{\partial x^{(t)}}\right)^{-T} \left(\delta x^{(t-1)} + \Delta t \frac{\partial M}{\partial p} \delta p\right) \qquad (t = 1, \cdots, T)$$

$$\begin{cases} \delta\lambda^{(T+1)} = 0, \delta\nu^{(T+1)} = 0 \\ \delta\lambda^{(t)} = \left(I - \Delta t \frac{\partial M}{\partial x^{(t)}}\right)^{-T} \left\{\delta\lambda^{(t+1)} + \Delta t \left(\frac{\partial^2 M}{\partial {x^{(t)}}^2} \delta x^{(t)} + \frac{\partial^2 M}{\partial p \partial x^{(t)}} \delta p\right)^T \lambda^{(t)} + \frac{\partial^2 \tilde{J}}{\partial {x^{(t)}}^2} \delta x^{(t)}\right\} (t = T, \cdots, 1) \\ \delta\nu^{(t)} = \delta\nu^{(t+1)} + \Delta t \left(\frac{\partial^2 M}{\partial p \partial x^{(t)}} \delta x^{(t)} + \frac{\partial^2 M}{\partial p^2} \delta p\right)^T \lambda^{(t)} + \Delta t \left(\frac{\partial M}{\partial p}\right)^T \delta\lambda^{(t)} \qquad (t = T, \cdots, 1), \end{cases}$$

$$\frac{\partial^2 J}{\partial {\theta^{(0)}}^2} \delta\theta^{(0)} = \begin{pmatrix} \delta\lambda^{(1)} + \frac{\partial^2 \tilde{J}}{\partial {x^{(0)}}^2} \delta x^{(0)} \\ \delta\nu^{(1)} + \frac{\partial^2 L}{\partial p^2} \delta p \end{pmatrix} \text{for } \forall \delta\theta^{(0)} = \left(\delta x^{(0)}\ \delta p\right)^T.$$

Thus, precision matrix $\frac{\partial^2 J}{\partial {\theta^{(0)}}^2}$ can be obtained calculating Hessian-vector product for $d\theta^{(0)} = (1 \ldots 0)^T, \ \ldots, (0 \ldots 1)^T$, and 1-sigma confidence interval (CI) can be calculated as

$$CI = diag\left(\sqrt{\left(\frac{\partial^2 J}{\partial {\theta^{(0)}}^2}\right)^{-1}}\right).$$

Moreover, $Corr\left(\theta^{(0)}\right)$, correlation coefficient for the estimated parameters and initial states, can be calculated as

$$Corr\left(\theta^{(0)}\right) = CI^{-1} \left(\frac{\partial^2 J}{\partial {\theta^{(0)}}^2}\right)^{-1} CI^{-1}.$$

Codes

model4DVar.py

ODE system with parameters.

discrete4DVar.py

Core Adjoint class for initial states $(x_0^{(0)}, \ldots, x_{N-1}^{(0)})^{\mathrm{T}}$ and parameters $(p_0, \ldots, p_{M-1})$ estimation.

discrete4DVarMain.py

Estimate initial states $(x_0^{(0)}, \ldots, x_{N-1}^{(0)})^{\mathrm{T}}$ and parameters $(p_0, \ldots, p_{M-1})$ given observed data file.

twinExperiment.py

First, generate observed data using given model, true initial states, and parameters with white noise. Then estimate initial states $(x_0^{(0)}, \ldots, x_{N-1}^{(0)})^{\mathrm{T}}$ and parameters $(p_0, \ldots, p_{M-1})$ given generated observed data. Estimated result is shown compared to the true initial state and parameters.

twinExperimentGivenData.py

Estimate initial states $(x_0^{(0)}, \dots, x_{N-1}^{(0)})^{\mathrm{T}}$ and parameters $(p_0, \dots, p_{M-1})$ given generated observed data file. Estimated result is shown compared to the true initial state and parameters.

twinExperimentIteration.py

Same as twinExperiment.py, but iteratively conduct estimation using randomly selected initial states.

util.py

Plotting utilities.

Example: modified Lorenz96 model

$$\frac{dx_i}{dt} = p_1(x_{i+1} - x_{i-2})x_{i-1} + p_0 \ (i = 0, \dots, N-1)$$

True parameters are set to be $p_0 = 8, p_1 = 1$.