

**Data Assimilation Program:
User Documentation**

by

David A. Bailey

March 2006

Introduction

Data assimilation aims to incorporate measured observations into a dynamical system model in order to produce accurate estimates of all the current (and future) state variables of the system. The optimal estimates minimize a variational principle and can be found using adjoint methods. The model equations are treated as strong constraints on the problem. In reality, the model does not represent the system behaviour exactly and errors arise due to lack of resolution and inaccuracies in physical parameters, boundary conditions and forcing terms. In [2] the authors described a technique for estimating systematic and time correlated errors as part of the variational assimilation procedure. The modified method determines a correction term that compensates for the model error and leads to improved predictions of the system states. This program illustrates the technique through application to a system which is derived from a standard explicit finite difference approximation to the heat equation

$$v_t = \sigma v_{zz} + s(z). \quad (1)$$

The aim of the new technique is to estimate the systematic, time-correlated components of the model error along with the dynamical model states as part of the variational assimilation procedure. Although the general form of the model error is not known, some simple assumptions about the evolution of the model error can be made. An augmented system for both model states and model error is derived in [2] and detailed below. The control variables are reduced to the unknown initial values of the model states and the model errors, and the corresponding optimization may be solved efficiently. A comprehensive development of the technique is presented, together with applications and results, in [2].

The Augmented Data Assimilation Problem

The augmented system of equations for the model states and model error is given by

$$\mathbf{x}_{k+1} = \mathbf{f}_k + T_k \mathbf{e}_k, \quad (2)$$

$$\mathbf{e}_{k+1} = \mathbf{g}(\mathbf{x}_k, \mathbf{e}_k), \quad (3)$$

$k = 0, \dots, N - 1$, where $\mathbf{x}_k \in \mathbb{R}^n$ is the model state at time t_k , $\mathbf{f}_k : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is a non-linear function describing the evolution of the state from time t_k to time t_{k+1} , $\mathbf{e}_k \in \mathbb{R}^m$ represents the systematic components of the model error, matrix T_k is a prescribed matrix that defines the distribution of the systematic error terms \mathbf{e}_k in the model error, and $\mathbf{g} : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^m$ is a function which describes the evolution of the systematic error terms. In practice very little is known about the form of the model error and a simple form of the error evolution that reflects any available knowledge needs to be specified.

The observations are related to the system states by the equations

$$\mathbf{y}_k = \mathbf{h}_k(\mathbf{x}_k) + \delta_k, \quad (4)$$

$k = 0, \dots, N - 1$, where $\mathbf{y}_k \in \mathbb{R}^{p_k}$ is a vector of p_k observations at time t_k and $\mathbf{h}_k : \mathbb{R}^n \rightarrow \mathbb{R}^{p_k}$ is a non linear function that includes transformations and grid interpolations. The observational errors $\delta_k \in \mathbb{R}^{p_k}$ are assumed to be unbiased, serially uncorrelated, Gaussian random vectors with known covariance matrices $R_k \in \mathbb{R}^{p_k \times p_k}$.

It is also assumed that prior estimates, or 'background estimates', \mathbf{x}_0^b and \mathbf{e}_0^b of \mathbf{x}_0 and \mathbf{e}_0 , respectively, are known and that the covariance matrices of the errors ($\mathbf{x}_0 - \mathbf{x}_0^b$) and ($\mathbf{x}_0 - \mathbf{x}_0^b$) are given by $B_0 \in \mathbb{R}^{n \times n}$ and $Q_0 \in \mathbb{R}^{m \times m}$. The observational errors and the errors in the prior estimates are not correlated.

The aim of the data assimilation is to minimize the square errors between the model predictions and the observed system states, weighted by the inverse of the covariance matrices, over the assimilation interval. The augmented system equations (3) are treated as strong constraints on the problem. The initial values \mathbf{x}_0 and \mathbf{e}_0 of the model state and model error completely determine the response of the augmented system and are taken, therefore, to be the control variables in the optimization. The problem is well-posed, in general, if the square errors between the prior estimates and the control variables are included in the objective function. The data assimilation problem is thus given by:

Minimize, with respect to the \mathbf{x}_0 and \mathbf{e}_0 , the objective function

$$\begin{aligned} \mathcal{J} = & \frac{1}{2} (\mathbf{x}_0 - \mathbf{x}_0^b)^T B_0^{-1} (\mathbf{x}_0 - \mathbf{x}_0^b) + \frac{1}{2} \sum_{j=0}^{N-1} (\mathbf{h}_j(\mathbf{x}_j) - \mathbf{y}_j)^T R_j^{-1} (\mathbf{h}_j(\mathbf{x}_j) - \mathbf{y}_j) \\ & + \frac{1}{2} (\mathbf{e}_0 - \mathbf{e}_0^b)^T Q_0^{-1} (\mathbf{e}_0 - \mathbf{e}_0^b), \end{aligned} \quad (5)$$

subject to the augmented system (3).

The constrained minimization problem can again be converted into an unconditioned problem using the method of Lagrange. Necessary conditions for a solution to the unconstrained problem (5) require that a set of adjoint equations together with the system equations must be satisfied. The adjoint equations are given by

$$\boldsymbol{\lambda}_N = 0, \quad (6)$$

$$\boldsymbol{\mu}_N = 0, \quad (7)$$

and

$$\boldsymbol{\lambda}_k = F_k^T(\mathbf{x}_k) \boldsymbol{\lambda}_{k+1} + G_k^T(\mathbf{x}_k, \mathbf{e}_k) \boldsymbol{\mu}_{k+1} - H_k^T R_k^{-1} (\mathbf{h}_k(\mathbf{x}_k) - \mathbf{y}_k), \quad (8)$$

$$\boldsymbol{\mu}_k = T_k^T \boldsymbol{\lambda}_{k+1} + \Gamma_k^T(\mathbf{x}_k, \mathbf{e}_k) \boldsymbol{\mu}_{k+1}, \quad (9)$$

for $k = N - 1, \dots, 0$, where $\boldsymbol{\lambda}_k \in \mathbb{R}^n$, $\boldsymbol{\mu}_k \in \mathbb{R}^m$ are the adjoint variables, $F_k^T \in \mathbb{R}^{n \times n}$, $H_k^T \in \mathbb{R}^{n \times p_k}$ and $G_k^T \in \mathbb{R}^{m \times n}$ are the Jacobians of \mathbf{f}_k , \mathbf{h}_k and \mathbf{g}_k with respect to \mathbf{x}_k , respectively, and $\Gamma_k \in \mathbb{R}^{m \times m}$ is the Jacobian with respect to \mathbf{e}_k .

The gradients of the objective function (5) with respect to the initial data \mathbf{x}_0 and \mathbf{e}_0 are then given by

$$\nabla_{\mathbf{x}_0} \mathcal{J} = B_0^{-1} (\mathbf{x}_0 - \mathbf{x}_0^b) - \boldsymbol{\lambda}_0, \quad (10)$$

$$\nabla_{\mathbf{e}_0} \mathcal{J} = Q_0^{-1} (\mathbf{e}_0 - \mathbf{e}_0^b) - \boldsymbol{\mu}_0. \quad (11)$$

For the optimal it is required that the gradients (10) and (11) be equal to zero. Otherwise these gradients provide the local descent direction needed by the iteration procedure to find an improved estimate for the optimal initial values of the augmented system. In each step of the gradient iteration the augmented equations are solved in the forward direction, starting from the current best estimate of the initial conditions, and the corresponding adjoint equations are solved in the reverse direction. The estimated initial values are then updated using the computed gradients.

The Program

The program is designed to examine the performance of data assimilation with the augmented system, when a constant bias error correction is applied

$$\mathbf{e}_{k+1} = \mathbf{e}_k, \quad T_k = \mathbf{I}. \quad (12)$$

This choice allows for a constant vector $\mathbf{e} = \mathbf{e}_0$ of unknown ‘dynamical parameters’ to be found. This form is expected to be appropriate for representing average errors in source terms or in boundary conditions.

The ‘true’ system modelled is derived from a standard explicit finite difference approximation to the heat equation

$$v_t = \sigma v_{zz} + s. \quad (13)$$

with zero boundary conditions at $z = 0, 1$ and a point source of the form

$$s = s(z) = \frac{1}{3} \delta(z - x_p) \quad (14)$$

or

$$s = s(t, z) = \frac{\sin(t)}{3} \delta(z - x_p) \quad (15)$$

where δ denotes the Dirac delta function and x_p is the spatial position of the source. The model equations are given by

$$x_j^{k+1} = x_j^k + \sigma \Delta t (x_{j-1}^k - 2x_j^k + x_{j+1}^k) / \Delta z^2 + \Delta t s_j, \quad (16)$$

$$x_0^{k+1} = 0, \quad (17)$$

$$x_J^{k+1} = 0, \quad (18)$$

with $j = 1, 2, \dots, J-1$, $k = 0, 1, \dots, N-1$, and where the model variables x_j^k approximate $v(j\Delta z, k\Delta t)$ with $\Delta t = (1/N)$ and $\Delta z = (1/J)$. The dimension of the system is n , where $n = J-1$. Equation (16) is a ‘ θ -method’ discretisation of equation (??), with $\theta = 0$.

The discretisation of the source term (14) is given by

$$s_j = \begin{cases} \frac{1}{3\Delta z} & j = p \\ 0 & \text{otherwise} \end{cases}, \quad (19)$$

where p is the index of the spatial node where the point source is positioned [1]. The discretisation of source term (15) is given by

$$s_{k,j} = \begin{cases} \frac{\sin(k\Delta t)}{3\Delta z} & j = p \\ 0 & \text{otherwise} \end{cases}. \quad (20)$$

The observations are taken from the ‘true’ solution states of discrete equations (16) - (18). The position of the observation are assumed to coincide with the finite difference mesh, with

$$\mathbf{h}_k(\mathbf{x}_k) = C\mathbf{x}_k, \quad (21)$$

where the matrix $C \in \mathbb{R}^{p_k \times n}$ is constructed as follows: if an observation exists at the spatial node with index I then $C_{I,I} = 1$, otherwise the entries in the matrix are zero.

For example, if $J = 6$ and observations are taken at x_1, x_2 and x_5 then

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & \dots \\ 0 & 1 & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & 1 & \dots \end{pmatrix}. \quad (22)$$

By definition, the entries of $F_k(\mathbf{x}_k)$ are given by

$$F_{i,i} = 1 - 2\nu, \quad (23)$$

$$F_{i,i-1} = \nu, \quad (24)$$

$$F_{i,i+1} = \nu, \quad (25)$$

where $\nu = \sigma \Delta t / \Delta z^2$. In addition, $G_k(\mathbf{x}_k, \mathbf{e}_k) = 0$, $H_k = C$ and $\Gamma = \mathbf{I}$.

The covariances of the prior estimates and of the observations are taken to be such that $B_0^{-1} = 0$, $Q_0^{-1} = q\mathbf{I}$ and $R_k^{-1} = (2/N)\mathbf{I}$, $\forall k$.

The significance of choosing $B_0^{-1} = 0$ is that no prior estimates of the initial values of the state variables are assumed to be known, which implies that there are no constraints on the initial data that can be selected in the minimization process. The observations available are sufficient to ensure that the variational problem remains well-posed under this assumption. Thus, if there were no error on the initial data used to generate the ‘true’ states from which the observations are taken and if there were no error present in the model, then the data assimilation process should reproduce the correct initial data.

The assimilation procedure is performed on an ‘imperfect’ system model, which is derived from the ‘true’ system model (16) - (18) by either the omission of the source term(s) and/or the use of an alternative source term(s). The aim of the assimilation being to estimate the ‘true’ system using the observations and the ‘imperfect’ model.

The minimisation is solved using a conjugate gradient solver. The convergence criterion for the iteration is given by $\|\nabla_{\mathbf{u}}\mathcal{J}\| \leq 10^{-6}$, where \mathbf{u} represents the control variables and $\|\cdot\|$ denotes the L_2 - norm.

The forecast is performed starting from the end of the assimilation interval.

Downloading and Compiling the Code

The following files need to be downloaded:

- setup.f90 - Fortran 90 source code for declaring constant parameters and global variables.
- theta_method.f90 - Fortran 90 source code for evaluating the θ -method finite difference scheme solution to heat equation (13).
- CALFG.f90 - Fortran 90 source code for evaluating the objective function (5) and its gradients (10) and (11).
- conmin.f90 - Fortran 90 source code for solving the minimization problem using the CONMIN conjugate gradient method [3].

- data_assim.f90 - Fortran 90 source code for the main program unit.
- input_data.dat - Input file for the program. When running, the program will read information from this file. It may be edited using a text editor.

A Fortran 90 compiler is required for compiling the code. For suitably equipped UNIX systems, the compilation is achieved by

```
f90 setup.f90 theta_method.f90 CALFG.f90 conmin.f90 data_assim.f90 -o assim.out
```

which names the executable as ‘assim.out’. Note: (1) the file ‘input_data.dat’ is NOT compiled, and (2) no special libraries are required.

If you have been editing the code with a Windows program you may need to run your source code through the utility **dos2unix** to prepare the formatting for the UNIX compiler. This achieved by typing

```
dos2unix -ascii filename.f90 filename.f90
```

where *filename* is the name of the file to re-format.

Program Input

Input to the program is controlled through the file ‘input_data.dat’. The user has the ability to set the following program components by altering the contents of the file ‘input_data.dat’ through the use of a text editor.

- σ - The thermal diffusivity constant in equation (13).
- The spatial domain $[x_{min}, x_{max}]$ in which the problem is solved.
- The number of solution nodes positioned inside the spatial domain - not including boundary nodes.
- The number, type and node position for any source term(s) in the ‘true’ systems.
- The initial value of the vector \mathbf{x}_0^b .
- The initial value of the vector \mathbf{e}_0^b .
- The end of time interval over which the assimilation is performed together with the number of temporal steps in that range.
- The end of time interval over which the forecast is performed together with the number of temporal steps in that range.
- Whether the forecast is performed with or without retaining the computed model error over the forecast interval.
- Whether or not the initial data \mathbf{x}_0 is used as control variables.
- The vector \mathbf{x}_0 .
- Whether or not the model error \mathbf{e}_0 is used as control variables.

- The vector \mathbf{e}_0 .
- The variable q in equation $Q_0^{-1} = q\mathbf{I}$.
- The number, type and node position for any source term(s) in the ‘imperfect’ and forecast systems.
- The number and node positions of any observations.
- Whether or not to generate the output file ‘graphx.m’.

The input file has the following format:

```

0.1  sigma :   w_t = sigma . w_xx + s1 + s2
1    1st source term for 'true' soln      ( [0] 0 [1] (1/3)*delta( x - S1*delta_x ) [2] (SIN(t)/3)*delta( x - S1*delta_x ) )
0    2st source term for 'true' soln      ( [0] 0 [1] (1/3)*delta( x - S2*delta_x ) [2] (SIN(t)/3)*delta( x - S2*delta_x ) )
0.0  x min
1.0  x max
15   number of internal spatial nodes (N)
4    spacial node position for the 1st source node
0    2nd source node
    x_0 background vector (nodes 1 to N):
      1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
    e_0 background vector (nodes 1 to N):
      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
40   number of temperal levels in assimilation interval
0.5  assimilation output time
40   number of temperal levels in forecast interval
1.0  forecast output time
0    forecast ( [0] without error [1] with error )
1    initial data as control variable      ( [0] true [1] false )
    x_0 vector (nodes 1 to N):
      1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
0    model error as control variable       ( [0] true [1] false )
    e_0 vector (nodes 1 to N):
      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0    background term variable q
0    1st source term for 'imperfect' model ( [0] 0 [1] (1/3)*delta( x - S1*delta_x ) [2] (SIN(t)/3)*delta( x - S1*delta_x ) )
0    2st source term for 'imperfect' model ( [0] 0 [1] (1/3)*delta( x - S2*delta_x ) [2] (SIN(t)/3)*delta( x - S2*delta_x ) )
0    1st source term for 'forecast' model  ( [0] 0 [1] (1/3)*delta( x - S1*delta_x ) [2] (SIN(t)/3)*delta( x - S1*delta_x ) )
0    2st source term for 'forecast' model  ( [0] 0 [1] (1/3)*delta( x - S2*delta_x ) [2] (SIN(t)/3)*delta( x - S2*delta_x ) )
5    number of observations
    observation vector (nodes 1 to N):
      0,0,0,1,1,0,1,0,0,1,0,1,0,0,0      ( [0] no observation [1] observation )
1    generate m-file      ( [0] No [1] Yes )

```

Figure 1: Format of file ‘input_data.dat’

Program Output

The program generates 2 output files; ‘output_data.dat’ and ‘graphx.m’. Upon completion of the program’s execution, the file ‘output_data.dat’ contains the following set of data

- The number of internal spatial nodes (the dimension of the system).
- The lower and upper bounds of the spatial domain upon which the problem is solved.
- The end of time interval over which the assimilation is performed.
- The end of time interval over which the forecast is performed.
- The number of observations.
- The position of the internal spatial nodes.
- The ‘true’, ‘imperfect’, assimilated and forecast model solution states at each temporal step of their respective time intervals.
- The model at each temporal step of the assimilation time interval.
- The position and value of any observations at each temporal step of the assimilation time interval.

The file ‘graphx.m’ is a MATLAB m-file that when compiled and run through MATLAB will create solution plots of the data contained within the file ‘output_data.dat’. The results are presented within 4 figures. Figures 1-3 show the ‘true’, ‘imperfect’ and assimilated model solution states together with the model error, at the beginning, middle and end of the assimilation interval. Figure 4 shows the ‘true’, ‘imperfect’ and forecast models at the end of the forecast interval. In all the figures a solid green line represents the solution to the ‘true’ system; the observation values, which are taken from the ‘true’ solution states, are denoted by black crosses; a red dot-dashed line represents the ‘imperfect’ (unassimilated) solution state; blue circles connected by a dotted blue line represents the solution to the assimilation problem; a black dotted line denotes the model error values; and a blue dashed line (no circles) represents the forecast solution state.

The file ‘graphx.m’ may be modified with the use of a text editor. For example, the user may wish to change axis parameters, graph and axis labels and/or the output time for each set of model states that are displayed graphically. The original ‘graphx.m’ file can be retrieved by simply running the code with the last input variable in the file ‘input_data.dat’ set to true (i.e. 1).

Example Code Runs

The set-up and results of two experiments are presented: (1) where the initial data for the system are treated as unknown variables to be found by the assimilation procedure and the model error is not modelled, (2) where both the initial data and the model error are treated as control variables in the assimilation process.

Example 1

This problem was taken from [2]. The system model constitutes the heat equation with a thermal diffusivity constant $\sigma = 0.1$, a source term of the form (14) and initial data $\mathbf{x}_0^b = 1$.

The problem is solved on the spatial interval $[0, 1]$ using 15 internal spacial nodes, and with the source term positioned at the fourth internal node. The assimilation is

applied on the time interval $[0, 0.5]$ using 40 temporal steps. The forecast is performed on the time interval $[0.5, 1]$, starting from the assimilation solution at time $t = 0.5$, again using 40 temporal steps. By definition of the problem there is no model error to be retained over the forecast interval.

The initial data constitutes the control vector for the assimilation procedure, and is given a starting value $x_0 = 1$. The model error is not modelled, therefore it is not included as a control variable and $q = 0$. In both the ‘imperfect’ and forecast models the source term is omitted. Observations are taken each temporal step of the assimilation time interval at the fourth, fifth, seventh, tenth and twelfth internal spatial nodes. The format of the file ‘input_data.dat’ for this example problem is shown in Figure 1.

The graphical output depicting the results for the problem (created by running and compiling the file ‘graphx.m’) is shown in Figure 2. At the initial point the assimilation does not reproduce the ‘true’ initial state, but instead generates initial values that compensate for the model errors and ensures that the assimilated solution is as close as possible to the observations over the whole interval. The estimated state at the end of the assimilation interval ($t = 0.5$) is therefore closer to the ‘true’ state than the ‘imperfect’ (unassimilated) solution. The forecast from this position is still poor, however, due to the inaccuracy of the model.

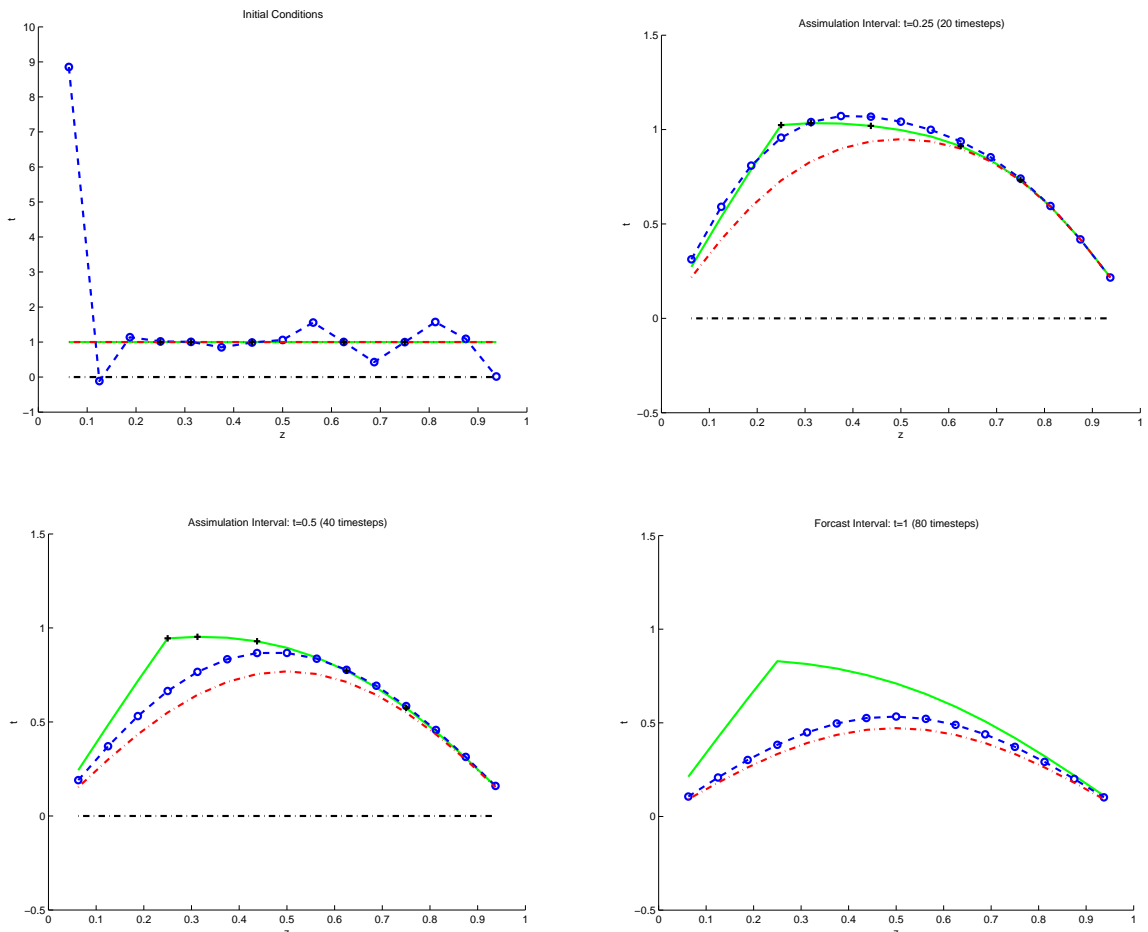


Figure 2: Example Problem 1

Example 2

Example problem 2 was also published in [2]. The numerical set-up is the same as in the first problem, only in this scenario the model error is included in the control vector and is retained over the forecast interval. Figure 3 displays the ‘input_data.dat’ file for this problem.

The results pertaining to this problem are shown in Figure 4. In this case the assimilated solution exactly matches the ‘true’ solution on the entire assimilation interval. In addition, retaining the computed model error correction over the forecast interval produces a perfect forecast.

```
0.1  sigma :   w_t = sigma . w_xx + s1 + s2
1    1st source term for 'true' soln      ( [0] 0 [1] (1/3)*delta( x - S1*delta_x ) [2] (SIN(t)/3)*delta( x - S1*delta_x ) )
0    2st source term for 'true' soln      ( [0] 0 [1] (1/3)*delta( x - S2*delta_x ) [2] (SIN(t)/3)*delta( x - S2*delta_x ) )
0.0  x min
1.0  x max
15   number of internal spatial nodes (N)
4    spacial node position for the 1st source node
0    2nd source node
x_0  background vector (nodes 1 to N):
      1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
e_0  background vector (nodes 1 to N):
      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
40   number of temperal levels in assimilation interval
0.5  assimilation output time
40   number of temperal levels in forecast interval
1.0  forecast output time
1    forecast ( [0] without error [1] with error )
1    initial data as control variable      ( [0] true [1] false )
x_0  vector (nodes 1 to N):
      1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
1    model error as control variable      ( [0] true [1] false )
e_0  vector (nodes 1 to N):
      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0    background term variable q
0    1st source term for 'imperfect' model ( [0] 0 [1] (1/3)*delta( x - S1*delta_x ) [2] (SIN(t)/3)*delta( x - S1*delta_x ) )
0    2st source term for 'imperfect' model ( [0] 0 [1] (1/3)*delta( x - S2*delta_x ) [2] (SIN(t)/3)*delta( x - S2*delta_x ) )
0    1st source term for 'forecast' model  ( [0] 0 [1] (1/3)*delta( x - S1*delta_x ) [2] (SIN(t)/3)*delta( x - S1*delta_x ) )
0    2st source term for 'forecast' model  ( [0] 0 [1] (1/3)*delta( x - S2*delta_x ) [2] (SIN(t)/3)*delta( x - S2*delta_x ) )
5    number of observations
observation vector (nodes 1 to N):
      0,0,0,1,1,0,1,0,0,1,0,0,1,0,0,0      ( [0] no observation [1] observation )
0    generate m-file      ( [0] No [1] Yes )
```

Figure 3: Format of file ‘input_data.dat’ for Example problem 2.

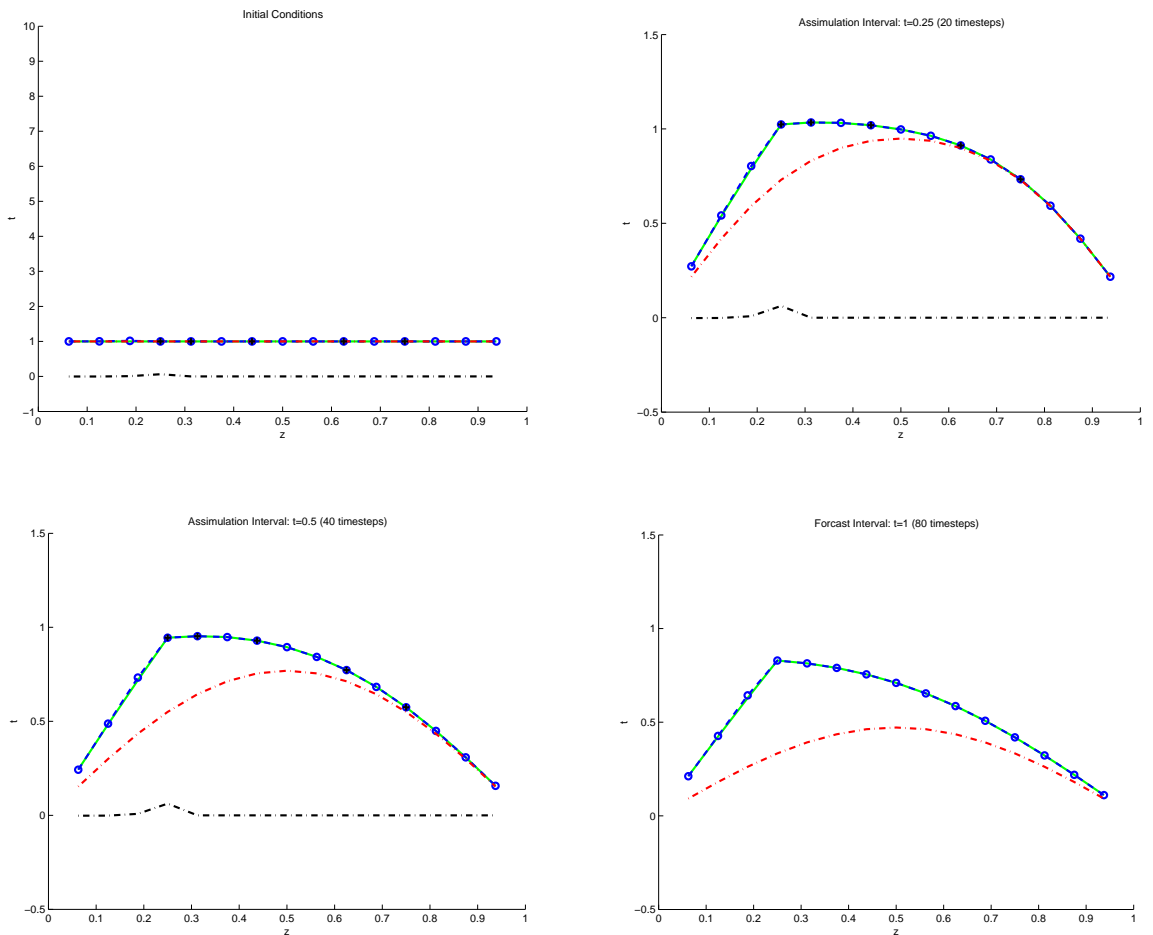


Figure 4: Example Problem 1

Notes for use

The code was written by David A. Bailey. The software may be used for education and private study purposes only. Please let me know if you have found any bugs in the code (bailey.da@gmail.com).

References

- [1] A. K. Griffith. Data Assimilation for Numerical Weather Prediction Using Control Theory. Phd Thesis, The University of Reading, 1997.
- [2] A. K. Griffith and N. K. Nichols. Adjoint Methods in Data Assimilation for Estimating Model Error. *Flow, Turbulence and Combustion*, **65**: 496-488, 2000.
- [3] <http://gams.nist.gov/serve.cgi/ModuleComponent/8535/Fullsource/NETLIB/500>