Chapter 7

Combos: Advection-Diffusion and operator splitting

Up to this point we have treated advection and diffusion as two separate processes and considered appropriate numerical techniques for solving each process. However, in many problems of interest both transport and diffusion occur simultaneously and we would like to be able to use the previously discussed methods to solve these more general problems. Unfortunately, as we have seen, a good technique for advection may not work for diffusion and vice versa (the simplest example is that diffusion equations are unstable if differenced using a staggered-leapfrog scheme while advection is unstable if solved with a FTCS scheme). This section will consider some additional techniques for solving coupled advection-diffusion problems and compare their accuracy.

In this section we will consider solutions to the 1-D scaled equations for the advection and diffusion of heat

$$\frac{\partial T'}{\partial t'} + \mathrm{Pe}\frac{\partial T'}{\partial z'} = \frac{\partial^2 T'}{\partial {z'}^2} \tag{7.0.1}$$

where T' is the dimensionless temperature and $\mathrm{Pe} = w_0 d/\kappa$ is the Peclet number. Because we will deal with strongly diffusive processes in this section, I have scaled the equations by the diffusion time $t = (d^2/\kappa)t'$. For strongly advective problems where $\mathrm{Pe} >> 10$, it makes more sense to scale the equations by the advection time $t = (d/w_0)t''$ (i.e. $t'' = \mathrm{Pe}t'$), in which case the equation becomes

$$\frac{\partial T'}{\partial t''} + \frac{\partial T'}{\partial z'} = \frac{1}{\mathrm{Pe}}\frac{\partial^2 T'}{\partial {z'}^2} \tag{7.0.2}$$

Either approach is valid and must give the same dimensional solutions.

For the initial condition of a gaussian distribution of temperature

$$T'(z', 0) = A \exp\left[-\frac{(z - z_0)^2}{\sigma^2}\right] \tag{7.0.3}$$

in an infinite medium, equation (7.0.1) has the convenient analytical solution

$$T'(z', t') = \frac{A}{\sqrt{1 + 4t'/\sigma^2}} \exp\left[-\frac{[(z - \mathrm{Pe}t') - z_0]^2}{\sigma^2 + 4t'}\right] \tag{7.0.4}$$

where A is the initial amplitude, z_0 is the initial position of the peak and σ is the original "half-width" (one standard-deviation). Thus in a perfect solution, the original bump will move off at a constant speed Pe and widen and decrease in amplitude. The question is how well do the various numerical schemes reproduce this result.

7.1 A smorgasbord of techniques

This section will compare several numerical schemes to the analytic solution (7.0.4). All of the problems will use $A = 2$ (over a background $T = 1$), $\mathrm{Pe} = 5$, $z_0 = 10$, $\sigma = 1$. The problem will be solved over a region $0 < z' < 50$ with 501 grid points $\Delta z = .1$, for a maximum time of $t' = 4$.

7.1.1 Explicit FTCS

The simplest scheme of all is a forward-time centered-space (FTCS) discretization of the full equation, i.e.

$$\frac{T_j^{n+1} - T_j^n}{\Delta t} = -Pe\frac{T_{j+1}^n - T_{j-1}^n}{2\Delta z} + \frac{T_{j+1}^n - 2T_j^n + T_{j-1}^n}{\Delta z^2} \tag{7.1.1}$$

or rearranging for T_j^{n+1} we get the simple updating scheme in stencil form

$$T^{n+1} = \left[\begin{array}{ccc} (\beta + \alpha) & (1 - 2\beta) & (\beta - \alpha) \end{array} \right] T^n \tag{7.1.2}$$

where $\alpha = \mathrm{Pe}\Delta t/(2\Delta z)$ is the local courant number and $\beta = \Delta t/\Delta z^2$ is the grid-space diffusion time. This scheme can be coded using something akin to

```
cm=beta+alpha
cc=1-2.*beta
cp=beta-alpha
do i=2,npnts-1
   ar1(i)= cm*ar2(i-1)+cc*ar2(i)+cp*ar2(i+1)
enddo
```

While FTCS is completely unstable for pure advection (because of negative numerical diffusion), it is stabilized by the addition of some real diffusion as long as the time step is smaller than the diffusive stability limit, i.e $\beta < 1/2$. Figure 7.1 compares the analytic solution and the calculated solution and their fractional error $\epsilon = T_{calc}/T_{true} - 1$ for the FTCS scheme. Because of the inherent negative diffusion, the calculated solutions do not decay as fast as they should and the overall error near the peak is positive and of order 5×10^{-3}. Which is not bad but it is noticeable (and we can do much better). In addition, the step size for a FTCS scheme must be very small and this solution requires 1600 time steps.

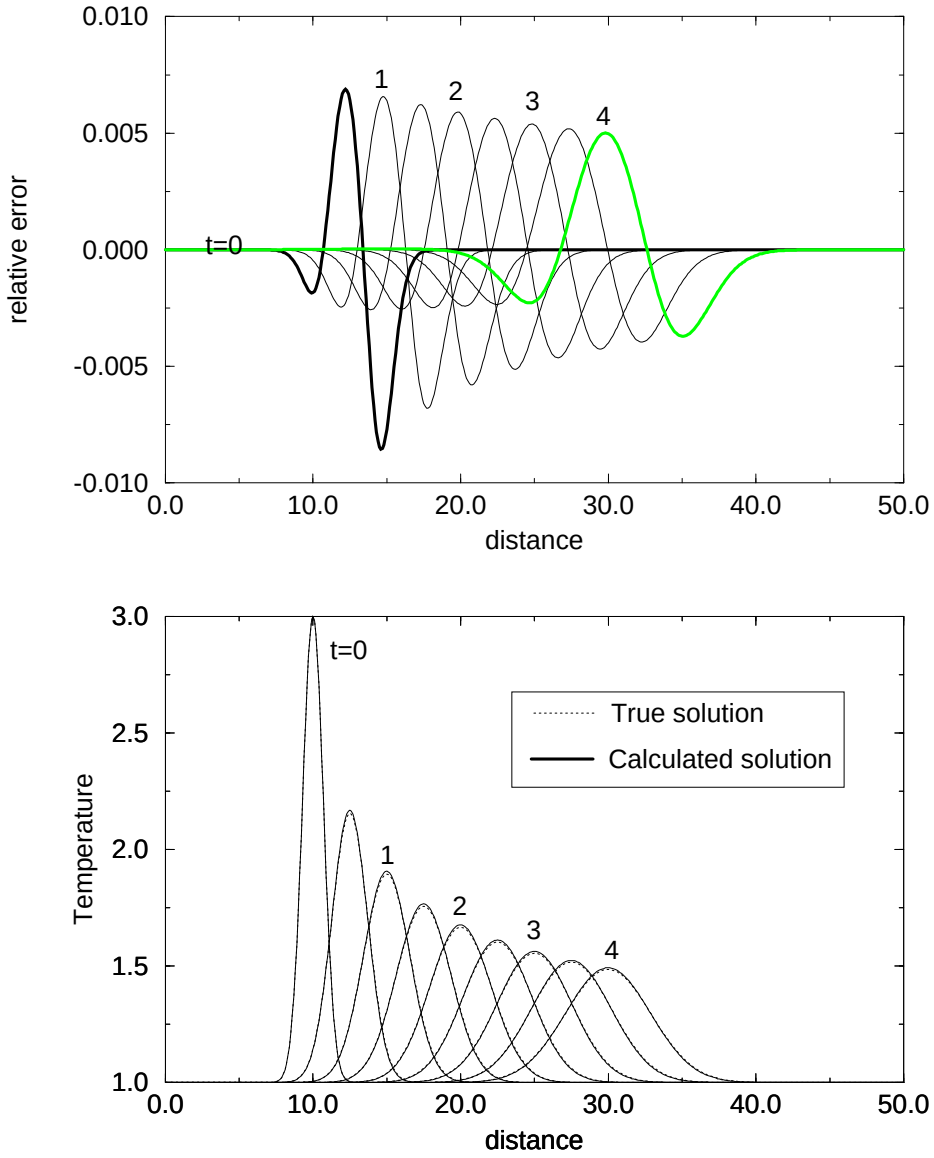

Figure 7.1: Solution and error of the model problem using an explicit FTCS differencing scheme.

7.1.2 Generalized Crank-Nicholson

The second scheme we will try is a more generalized version of the Crank-Nicholson scheme for diffusion which is implicit, stable for relatively large time steps and about a factor of 5 more accurate than the FTCS scheme. If we consider the FTCS scheme as a general operator equation of the form

$$T_j^{n+1} = T_j^n + \mathcal{L}T_j^n \tag{7.1.3}$$

where $\mathcal{L}$ is an operator that conveniently soaks up all the details, then a generalized crank nicholson scheme looks like

$$T_j^{n+1} = T_j^n + \frac{1}{2}\left[\mathcal{L}T_j^{n+1} + \mathcal{L}T_j^n\right] \tag{7.1.4}$$

as it is just the average of the rates at the present and future time. Using centered differences for the advection and diffusion parts of the operator, the implicit crank-nicholson formulation for advection-diffusion can be written in stencil form as

$$\left[\begin{array}{ccc} -(\beta/2+\alpha) & (1+\beta) & (\alpha-\beta/2) \end{array}\right] T^{n+1} = \left[\begin{array}{ccc} (\beta/2+\alpha) & (1-\beta) & (\beta/2-\alpha) \end{array}\right] T^n \tag{7.1.5}$$

Here $\alpha = \mathrm{Pe}\Delta t/(4\Delta z)$ and β is the same as before. This system of equations is tridiagonal and can be inverted using a subroutine such as Numerical recipes `tridag`. The relevant fortran snippets would look like

```
c---------set the matrix coefficients
      cm = -(alpha + .5*beta)           !sub-diagonal   T_(i-1)
      cc = (1.+beta)                    ! diagonal T_i
      cp = alpha - .5*beta              ! super diagonal T_(i+1)
      do i=1,npnts
        a(i)= cm
        b(i)= cc
        c(i)= cp
      enddo

c---------set the right hand side

      cm =  (alpha + .5*beta)           !sub-diagonal T_(i-1)
      cc = (1.-beta)                    ! diagonal T_i
      cp = .5*beta  - alpha             ! super diagonal T_(i+1)
      do i=2,npnts-1                    ! interior points
        d(i)=cm*tar(i-1)+cc*tar(i) + cp*tar(i+1)
      enddo

c-----magically patch up the boundary conditions

      YOUR CODE HERE

c--------invert the matrix
      call tridag(a,b,c,d,ar,npnts)
```

Figure 7.2 shows the results of this scheme which requires only 400 steps and does not exhibit negative numerical diffusion. Because of the asymmetry in the operator matrix, this scheme will probably become unstable for large values of Pe (I need to work this out or it can be extra credit for those inclined).

7.1.3 Operator Splitting, MPDATA/Semi-Lagrangian scehmes+ Crank Nicholson

For simple differencing schemes, it is straightforward to extend standard techniques to include both advection and diffusion. However, we would also like to

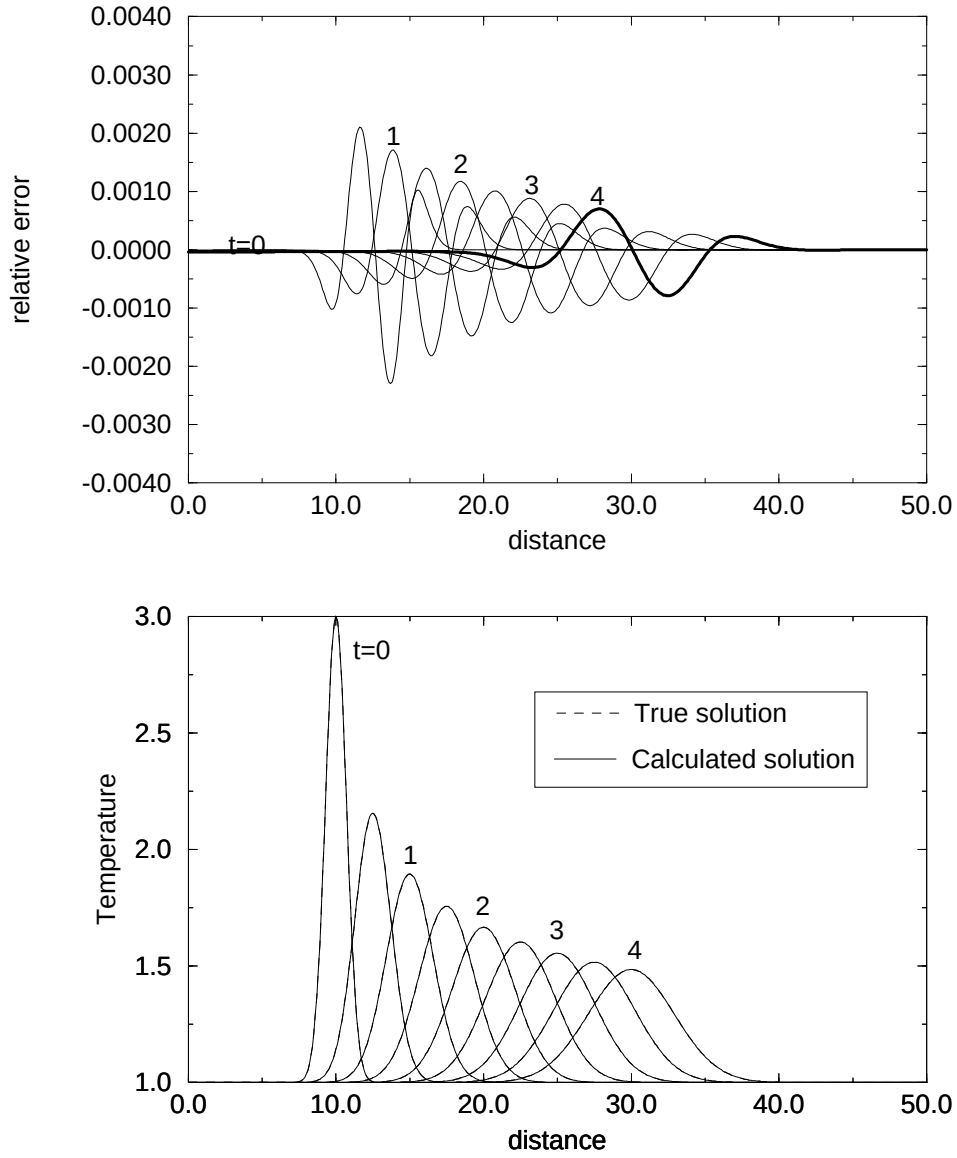

Figure 7.2: Solution and error of the model problem using an implicit crank-nicholson scheme.

be able to use something fancy like MPDATA or a semi-lagrangian scheme to handle the advection without having to rewrite the scheme to include diffusion[1]. In addition we would like to use a crank-nicholson scheme for just the diffusion part because it is highly accurate and unconditionally stable. The answer to our dilemma is a useful kludge known as *operator splitting* which simply suggests that we use each method for each process separately i.e. first advect and then diffuse (or vice versa).

[1]Although look at Section 7.2 as it is not actually much harder to do this with semi-lagrangian schemes

More formally, we can think of our initial equation as being of the form

$$\frac{\partial T}{\partial t} = \mathcal{L}_{adv}T + \mathcal{L}_{diff}T \tag{7.1.6}$$

where we have two (or more operators) that affect T in an additive way. Following Numerical recipes, we will now suppose that for each operator we have a good updating scheme that would take T^n to T^{n+1} if it were the only operative process. i.e. we could write the updatings as

$$\begin{aligned} T^{n+1} &= \mathcal{U}_{adv}(T^n, \Delta t) \\ T^{n+1} &= \mathcal{U}_{diff}(T^n, \Delta t) \end{aligned} \tag{7.1.7}$$

(we could also continue to add more processes and more updating schemes). Now all that operator splitting says is to proceed from time n to time $n+1$ by the following series of updates

$$\begin{aligned} T^{n+(1/2)} &= \mathcal{U}_{adv}(T^n, \Delta t) \\ T^{n+1} &= \mathcal{U}_{diff}(T^{n+(1/2)}, \Delta t) \end{aligned} \tag{7.1.8}$$

that is, advect the solution for a time-step Δt without any diffusion, then take this new solution and diffuse it for the same amount of time without advection (and repeat ad nauseum). Note that we have not advanced the time to $n+2$ however both processes have occurred for a time of Δt (as they should).

While it may seem that this approach is not hugely physical, it turns out in practice, that if the time-steps are relatively small and the separate updating schemes are sufficiently superior to any hybrid scheme, then any inaccuracies introduced in the sequential application of the operators are still much smaller than the inaccuracies of a scheme that is a compromise between methods. To illustrate this, Figures 7.3–7.5 show the results of this operator splitting approach using MPDATA for the advection scheme and Crank-Nicholson for the diffusion scheme. The relevant fortran would look something like

```
c-------set  tridiagonal matrix coefficients for CN step (alpha=0)
      call setcofcn1(a,b,c,npnts,0.,beta)

c------loop in time, doing an mpdata step then a CN step
      do n=1,nsteps
        t=t+dt*(n-1)
        call mpdata1(wp,T,npnts,ncor,i3rd,wk(1,1),wk(1,2)) !advect
        call setrhscn1(T,d,npnts,0.,ad)  !set the rhs with the current temp.
        call tridag(a,b,c,d,T,npnts)     ! diffuse ala CN step
      enddo
```

The only differences in Figures 7.3–7.5 are the number of corrections `ncor` used in the mpdata step. For `ncor=1`, (7.3) the advection is a standard donor-cell upwind differencing and is terribly over-diffusive. One mpdata correction (`ncor=2`, Fig. 7.4) however reduces the maximum error to less than 3×10^{-4} which is an *order of magnitude better* than the combined Crank-Nicholson step. Taking another correction (`ncor=3`, Fig. 7.5) reduces the error by another factor of

2. All of these improvements are at the expense of running time (and they can be severe) however, if high accuracy or very long runs are required it's nice to know that it works.

Figure 7.6 shows an operator splitting solution where the advective step is taken using a semi-lagrangian scheme. As expected, this scheme works at least as well as the most expensive mpdata scheme but is much faster. Trial and error showed that the minimum error for the problem was attained for a time-step that was 2.5 times the size of the critical stability step for the mpdata schemes (For comparison, Fig. 7.5 has a time step that is only 0.5 of the critical step-size). Thus for shear number of time steps, this scheme is a factor of five faster. The actual speed difference is larger than this because of the relative complexities of the mpdata scheme.

7.1.4 A caveat on operator splitting (and other numerical schemes)

There may be another deeper reason why operator splitting works so well for this model problem and that is because the analytic solution is inherently separable, i.e. advection and diffusion are completely uncoupled in this problem. To see this, note that this problem could also be solved by moving into a frame that is moving with the peak. In this frame the problem is only a diffusion problem that could be solved very well using a crank-nicholson scheme. Thus in the actual analytic solution, it is possible to calculate an arbitrary amount of diffusion and then advect the peak a long way and still get the right solution. Whether this method works with problems where the transport and diffusion cannot be separated is less clear.

The general caveat for comparison to analytic solutions is that quite often, the reason that an analytic solution exists is because the problems are linear, or separable and in some sense, special case solutions. Whether a numerical scheme that works for a linear solution (or constant coefficient equation) will carry over to more complex problems is never guaranteed. When in doubt, do a convergence test by changing the grid space and time step and see how much the solutions vary. Finally, do enough analysis to convince yourself (and others) that the solutions are doing what the equations say they should. Remember *You're not finished, until you can prove that your solution is reasonable.*

7.2 Another approach: Semi-Lagrangian Crank-Nicholson schemes

To be honest, there is always something vaguely fishy about operator-splitting because it is often not symmetric with respect to the processes being considered. I.e. should you advect then diffuse or diffuse then advect? It's not always obvious (although in this case it makes no difference). In a perfect world you would like a scheme that handles all the possible processes in one consistent step yet doesn't add any new artifacts. As it turns out, for the advection-diffusion problem, there is a combined method that has all these properties and can solve for any advection-diffusion problem from $\mathrm{Pe} = 0 \rightarrow \infty$ with no artifacts and is a simple modification to the schemes we already know. The place to begin is to note that we could write

Eq. (7.0.2) in terms of the material derivative as

$$\frac{DT}{Dt} = \frac{1}{\mathrm{Pe}}\frac{\partial^2 T}{\partial x^2} \tag{7.2.1}$$

Now normally we couldn't solve (7.2.1) using the method of characteristics because the right-hand side depends on temperature gradients and therefore requires information about temperature on other characteristics. However, we do know how to solve something like (7.2.1) when things aren't moving. In Chapter 6 we just used a Crank-Nicholson scheme. The only difference now between Eq. (7.2.1) and Eq. (6.1.1) is that the time derivative is in a frame following a piece of material. But this is the frame in which we solve semi-Lagrangian schemes so it should be possible to combine the two approaches. If we just define $T^n_{j'}$ as the temperature at the point j' which will move to point j in time Δt then we can difference (7.2.1) with a Crank-Nicholson scheme like

$$\frac{T^{n+1}_j - T^n_{j'}}{\Delta t} = \frac{1}{2\mathrm{Pe}\Delta x^2}\left(\begin{bmatrix} 1 & -2 & 1 \end{bmatrix} T^n_{j'} + \begin{bmatrix} 1 & -2 & 1 \end{bmatrix} T^{n+1}_j\right) \tag{7.2.2}$$

which is identical to the standard Crank-Nicholson scheme except that now the point where the old temperature and old diffusion are calculated is at point j' not at point j. Put another way, we are just assuming that diffusion is a source term for the semi-Lagrangian solver and we are going to assume that the change in temperature along the flow path is just the integral of the amount of diffusion experienced during travel. Either way, we can rearrange (7.2.2) into a standard Crank-Nicholson Scheme

$$\begin{bmatrix} -1 & (\gamma+2) & -1 \end{bmatrix} T^{n+1}_j = \begin{bmatrix} 1 & (\gamma-2) & 1 \end{bmatrix} T^n_{j'} \tag{7.2.3}$$

where

$$\gamma = \frac{2\mathrm{Pe}\Delta x^2}{\Delta t} = \frac{2\mathrm{Pe}}{\beta} \tag{7.2.4}$$

The only difference is that the RHS is now evaluated at the interpolation point j' not at the grid point j. The important feature of this scheme is that the advection is hidden and doesn't change the symmetry of the operator. Moreover it is rather convenient that both simple diffusion and interpolation are linear operators and therefore, you can exchange the order of operations and calculate the RHS on the grid and then interpolate the value of the RHS. I.e. first calculate

$$d_j = \begin{bmatrix} 1 & (\gamma-2) & 1 \end{bmatrix} T^n_j \tag{7.2.5}$$

then the right hand side is just $d_{j'}$. Inflow boundary conditions can be tricky though with this technique.

Inspection of Eq. (7.2.4) shows that if $\mathrm{Pe} \to \infty$ then problem becomes completely dominated by the diagonal terms and $\gamma T^{n+1}_j = \gamma T^n_{j'}$ or $T^{n+1}_j = T^n_{j'}$ which is just the semi-Lagrangian scheme. For small Pe the scheme looks like a Crank-Nicholson scheme (although it probably should be rescaled). In the case that locally $w = 0$ then it just behaves like a CN scheme. Figure 7.7 shows the behaviour of this scheme and it is equally accurate to the operator-splitting approach. This scheme however, is slightly more prone to round-off error and I find it best to solve this one in double precision.

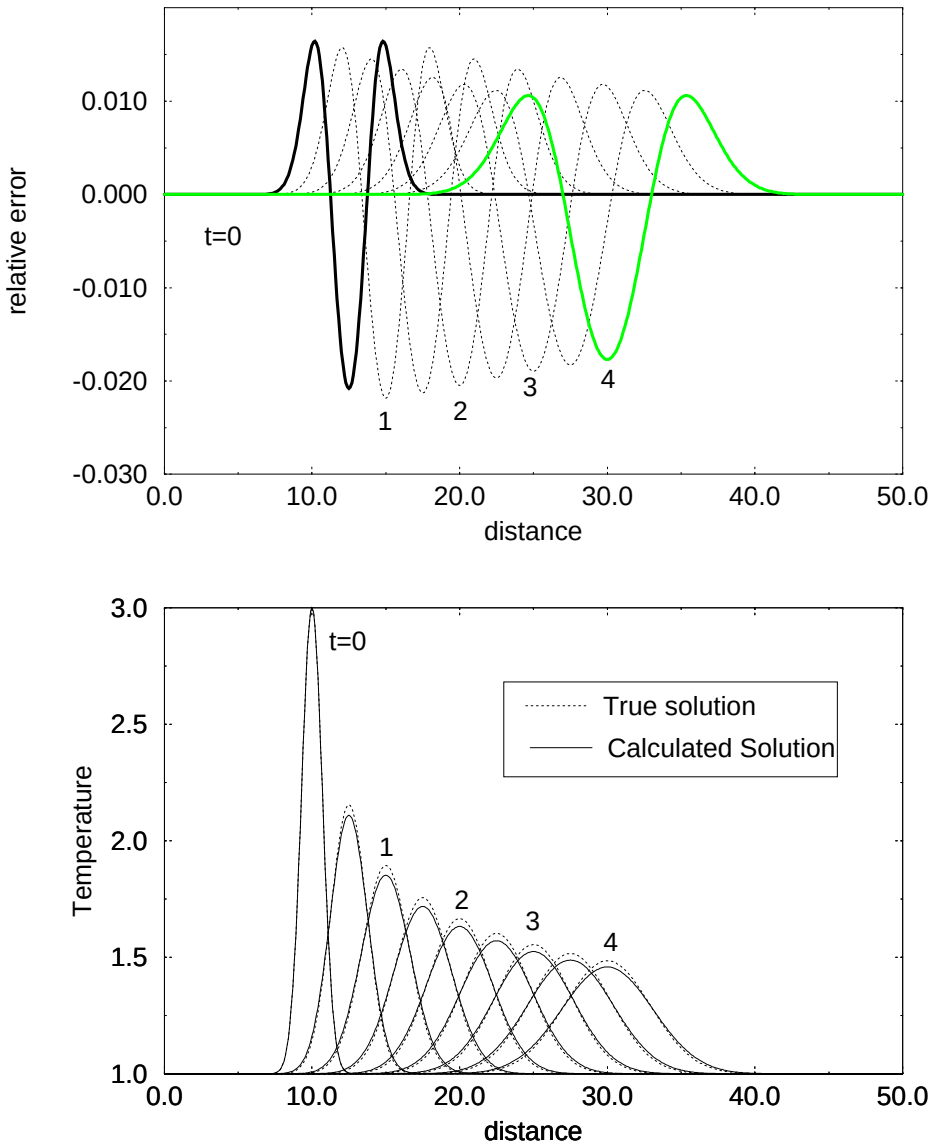

Figure 7.3: Solution and error of the model problem using operator splitting between a first order MPDATA (upwind-differencing, `ncor=1`) scheme for advection and a crank-nicholson scheme for diffusion. This problem has 400 steps. Note the extreme errors due to upwind differencing

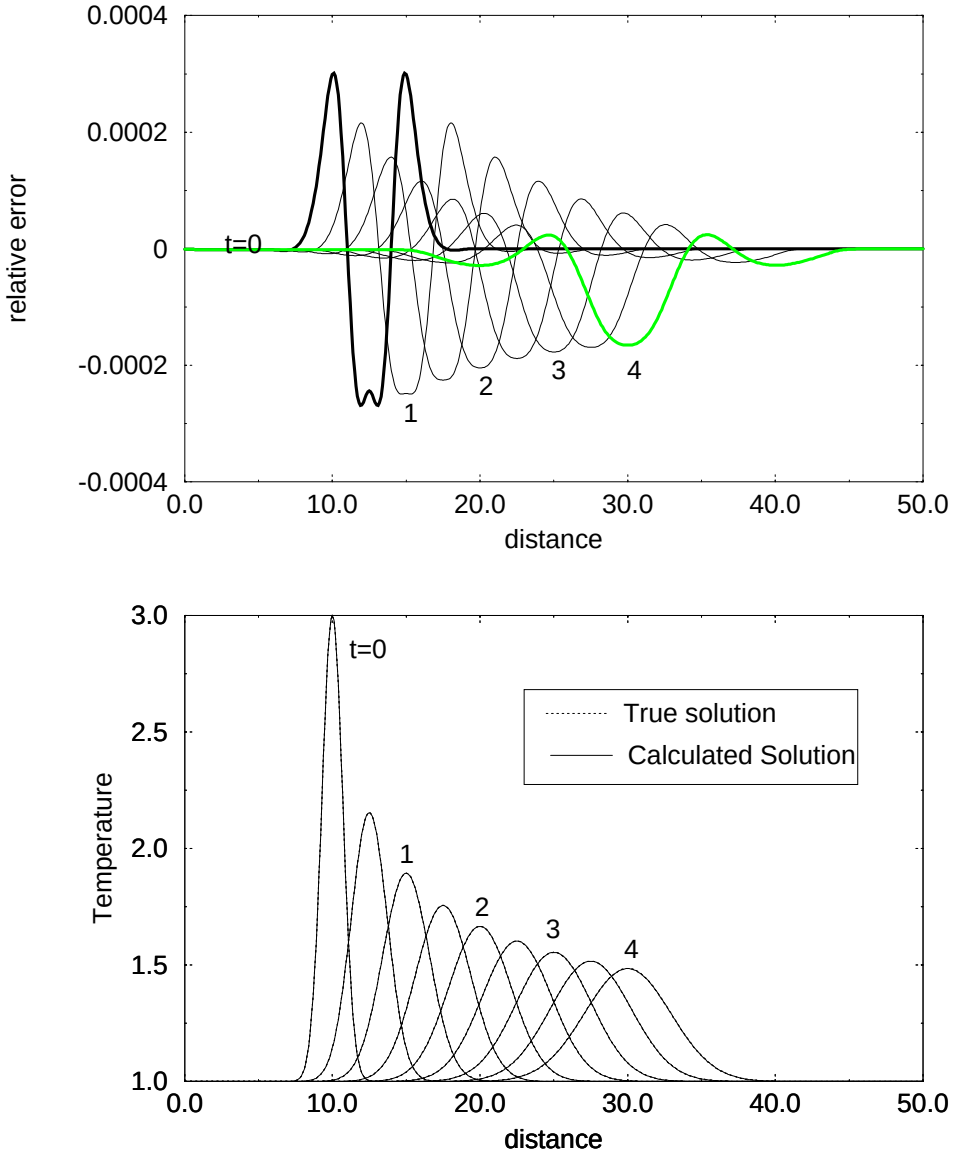

Figure 7.4: Same as Fig. 7.3 but with a second correction in MPDATA. Pretty amazing huh?. But see Fig. 7.6.

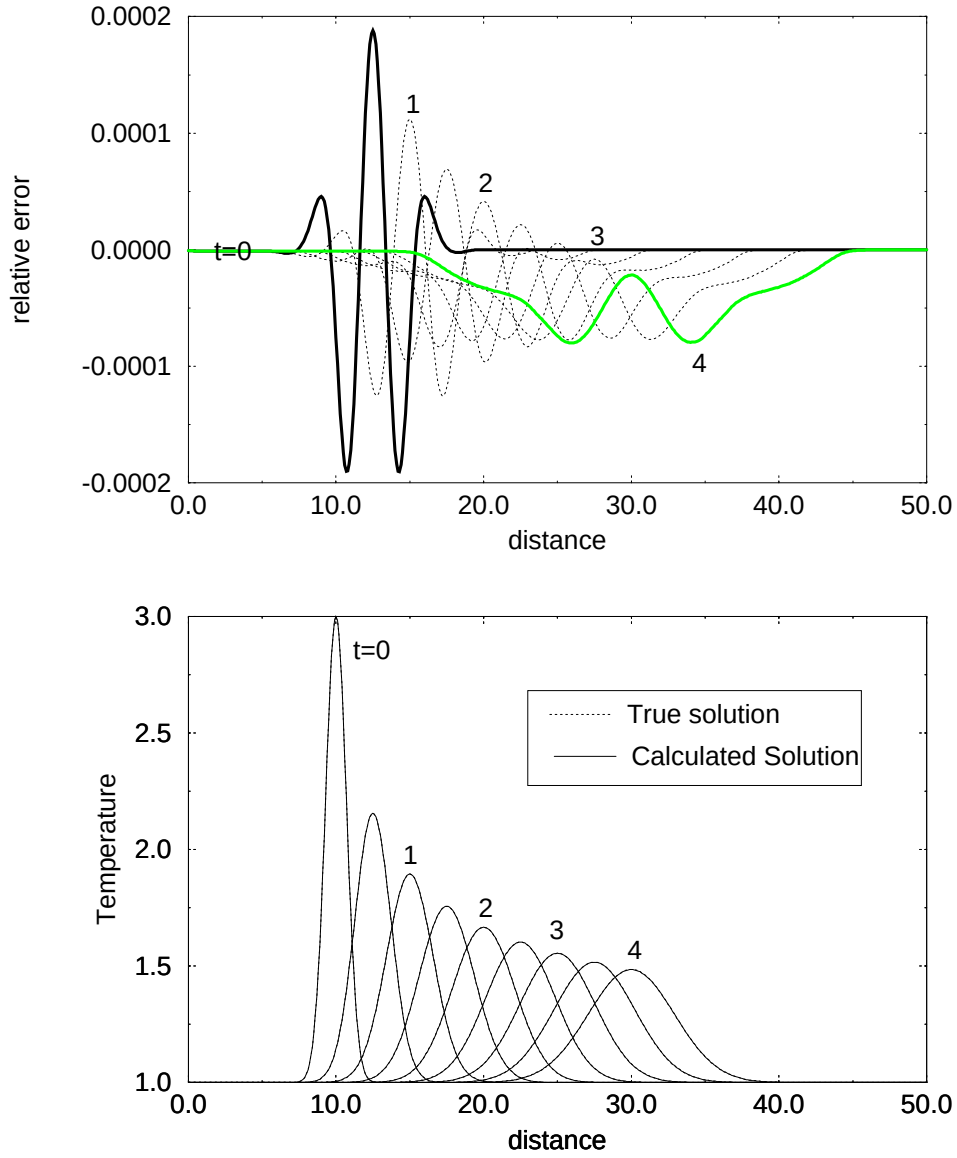

Figure 7.5: Same as Fig. 7.3 but with a third correction in MPDATA. Adding the anti-dispersive correction is almost indistinguishable as the diffusion removes the steep gradients that feed dispersion.

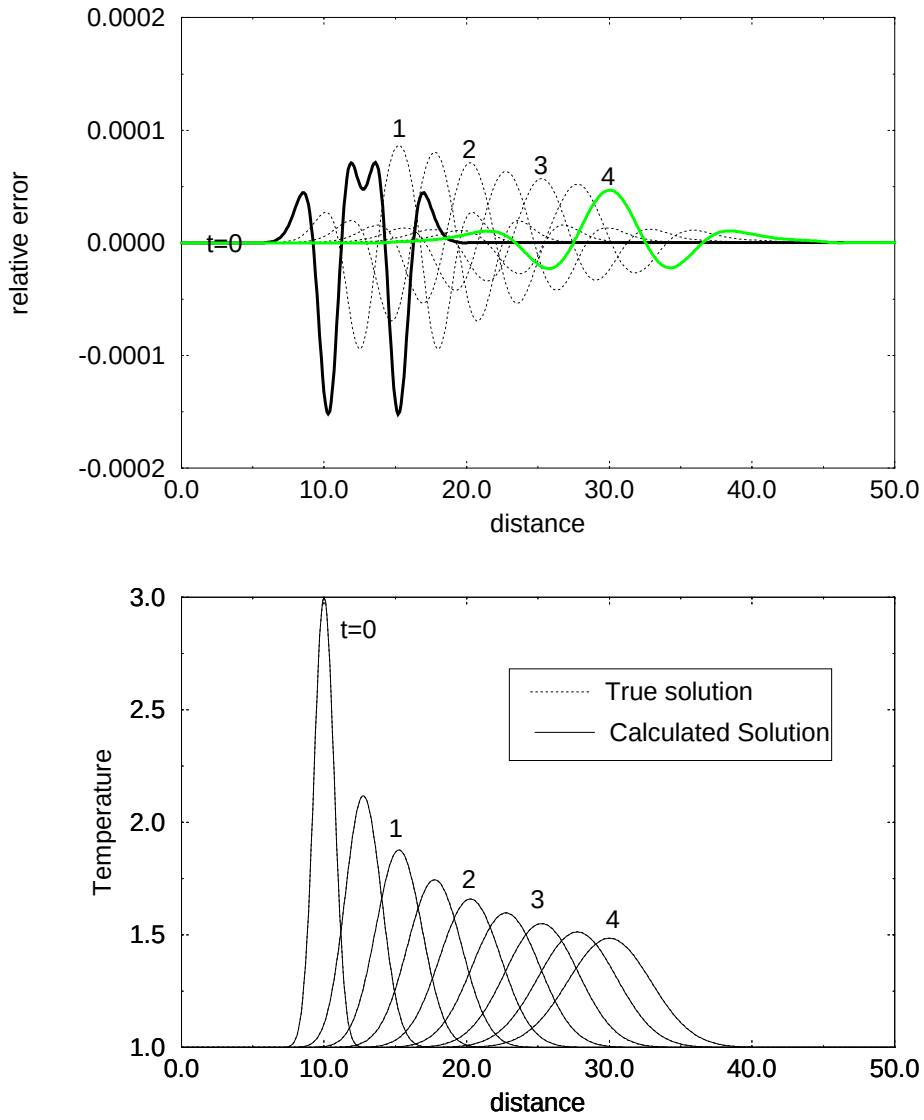

Figure 7.6: Same as Fig. 7.5 but now the advection step is done using a semi-lagrangian scheme with fractional stability condition $\alpha = 2.5$ (which seems to produce the smallest overall errors. Both larger and smaller step cause the error to increase). This scheme has comparable or smaller error than the best mpdata scheme uses 1/5th of the steps and is about an order of magnitude faster.

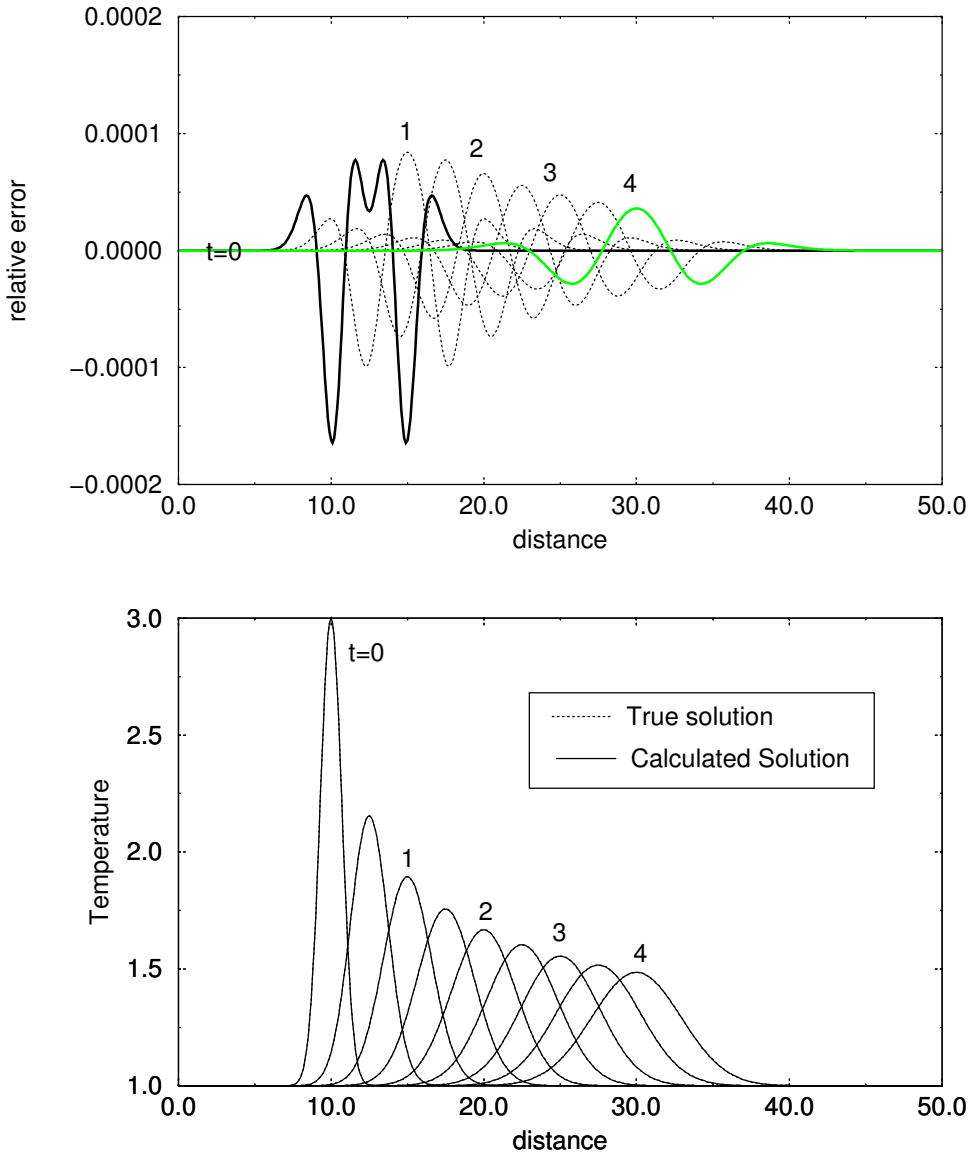

Figure 7.7: Results for the Semi-Lagrangian-Crank-Nicholson scheme which are identical to those for the SL-CN operator splitting problem (Fig. 7.6). This scheme, however, requires double precision to avoid some floating point drift engendered from the tridiagonal solvers.