Nonlinear Wavelet-Galerkin method

David Witman
Department of Scientific Computing

1 Definition of the Nonlinear problem

First we will remind ourselves of the difference between a linear and nonlinear problem. Equation 1 is a modified form of the one dimensional heat equation which is an initial boundary value problem. This is also a linear problem that we used in the introduction to the wavelet-Galerkin method

$$\begin{aligned} a_1\frac{\partial u}{\partial t} - a_2\frac{\partial^2 u}{\partial x^2} + a_3 u &= f(x,t) \text{ for } 0 < x < 1,\ 0 < t \leq T \\ u(0,t) = u(1,t) &= 0 \text{ for } 0 < t \leq T \\ u(x,0) &= u_0(x) \text{ for } 0 < x < 1 \end{aligned} \tag{1}$$

This problem is linear because we can establish a linear relationship between the solution values and the weak formulation defined. This equation will become nonlinear if the linear relationship is broken. So adding terms like u^2 or $u\frac{du}{dx}$ would mean that we can no longer linearly represent the weak form of this problem. Instead we must find a way to deal with these nonlinear terms in the context of the problem. For this work we will define an example nonlinear problem in equation 2 by adding the u^2 term to the modified heat equation.

$$\begin{aligned} a_1\frac{\partial u}{\partial t} - a_2\frac{\partial^2 u}{\partial x^2} + a_3 u + a_4 u^2 &= f(x,t) \text{ for } 0 < x < 1,\ 0 < t \leq T \\ u(0,t) = u(1,t) &= 0 \text{ for } 0 < t \leq T \\ u(x,0) &= u_0(x) \text{ for } 0 < x < 1 \end{aligned} \tag{2}$$

Using the same general process of developing our weak formulation in the linear case, it can be shown that the discrete weak formulation for the nonlinear modified heat equation is:

$$\begin{aligned} \sum_{j=1}^{n} C_j^k \left[a_1 \frac{1}{\Delta t}\int_0^1 \phi_j(x)\phi_i(x)dx + a_2\int_0^1 \phi_j'(x)\phi_i'(x)dx + a_3\int_0^1 \phi_j(x)\phi_i(x)dx \right] \\ + a_4\int_0^1 \left(\sum_{j=1}^{n} C_j^k \phi_j(x)\right)^2 \phi_i(x)dx = \int_0^1 f(x,t)\phi_i(x)dx + a_1\frac{1}{\Delta t}\int_0^1 u^{k-1}\phi_i(x)dx \end{aligned} \tag{3}$$

From this weak formulation we can clearly see that there is not a linear relationship between the unknown vector C_j^k and the scaling functions. This means that we must use a nonlinear method to solve this problem.

There are a number of different methods to resolve this type of problem including the bisection and Newton's method. Newton's method has a number of advantages over other methods like the bisection method including rapid convergence. But there are some disadvantages to Newton's

method; one of the biggest being: convergence is not guaranteed. To solve the nonlinear modified heat equation we will use Newton's method in the wavelet-Galerkin setting.

To perform a Newton iteration we hope to solve the linear system:

$$J(C_j^{k,n})(C_j^{k,n+1} - C_j^{k,n}) = -F(C_j^{k,n}) \tag{4}$$

where J represents the jacobian which accounts for all the derivatives of the residual vector, F. $C_j^{k,n}$ is the solution to our weak formulation at the n iteration, and so our only unknown ends up being $C_j^{k,n+1}$. In order to start the newton iteration off though we must first have a guess at the solution to get the iteration started. Often people using newton's iteration will use an initial guess of zeros or another constant. Since we have a time dependent problem we will use the solution at the previous time step as our initial guess since this will typically be closer to the actual solution than a constant value.

The first step in Newton's method is to define a function that you would like to find the roots of, or where the function is equal to zero. So for this example we will setup a residual function based on our weak formulation. This can be expressed in equation 5:

$$\begin{aligned} F(C_j^{k,n}) = &\int_0^1 f(x,t)\phi_i(x)dx + a_1\frac{1}{\Delta t}\sum_{j=1}^{n} C_j^{k-1,n}\int_0^1 \phi_i(x)dx \\ &- \sum_{j=1}^{n} C_j^{k,n}\left[a_1\frac{1}{\Delta t}\int_0^1 \phi_j(x)\phi_i(x)dx + a_2\int_0^1 \phi_j'(x)\phi_i'(x)dx + a_3\int_0^1 \phi_j(x)\phi_i(x)dx\right] \\ &- a_4\int_0^1 \left(\sum_{j=1}^{n} C_j^{k,n}\phi_j(x)\right)^2 \phi_i(x)dx \end{aligned} \tag{5}$$

Now that we have defined the residual function we must formulate our jacobian based on the partial derivative of our residual vector. Since the jacobian is based on the current newton iteration we must differentiate the residual with respect to $C_j^{k,n}$. This means that an arbitrary entry in our jacobian will take the form:

$$\begin{aligned} &\left[a_1\frac{1}{\Delta t}\int_0^1 \phi_j(x)\phi_i(x)dx + a_2\int_0^1 \phi_j'(x)\phi_i'(x)dx + a_3\int_0^1 \phi_j(x)\phi_i(x)dx\right] \\ &- 2a_4\int_0^1 \left(\sum C_j^{k,n}\phi_j(x)\right)\phi_j(x)\phi_i(x)dx \end{aligned} \tag{6}$$

we notice that the u^2 term is the only term where we need information about the solution to the newton iteration. From here there is one of two ways that we can proceed in the newton iteration: using three term connection coefficients to resolve the integral or use the sampling method method we used to approximate $f(x,t)$ to sample the newton iteration.

2 Three term Connection Coefficients

Solving for the three term connection coefficients is significantly more difficult than solving for the two term connection coefficients explained previously. In order to conceptualize the three term

connection coefficient problem we must first define the problem. We seek to determine integrals of the form:

$$\Omega_l^{d_1,d_2,d_3} evel = \int \phi_i^{d_1}(x)\phi_j^{d_2}(x)\phi_k^{d_3}(x)dx \tag{7}$$

which results in not a vector of connection coefficients, but a matrix of size $(2N-3)\times(2N-3)$, some of whose entries are zero. The elements that are nonzero can be found using the relation provided by Jordi Besora, assuming that scaling function $\phi_i^{d_1}(x)$ is fixed at zero:

$$2-N \leq j \leq N-2,\ 2-N \leq k \leq N-2 \text{ and } |j-k| \leq N-2 \tag{8}$$

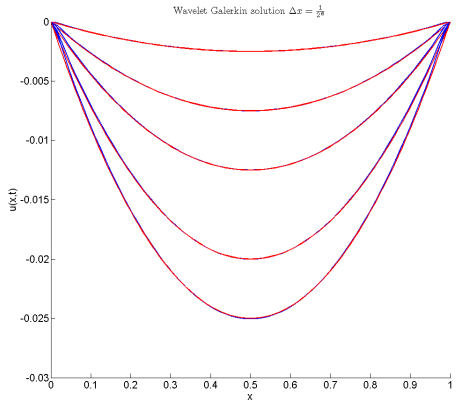
It can be shown that with these constraints, there are at most $3N^2-9N+7$ nonzero elements in the connection coefficient matrix. The system that one must solve takes the same form as the two-term connection coefficient problem except instead of a single moment condition, we must incorporate three. In order to compute a unique solution for this system, one must consider the moments for d_2, d_3 and the combination of d_2 and d_3. This makes the assumption that the derivative on the first function is zero. If it is not, a simple change of variables can put the problem into the form that we require.

For our problem we require integrals of the form:

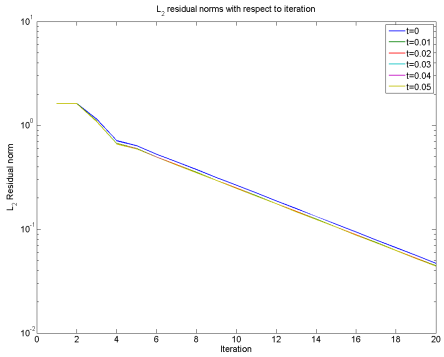
$$\Omega_l^{0,0,0} evel = \int \phi_j(x)\phi_j(x)\phi_i(x)dx \tag{9}$$

which makes things remarkably easier. First off we notice that we only have to worry about one shift since two of our three terms are the same $\phi_j(x)$ function. This means we set the single $\phi_i(x)$ function and then find the integrals with it against ϕ_j^2.

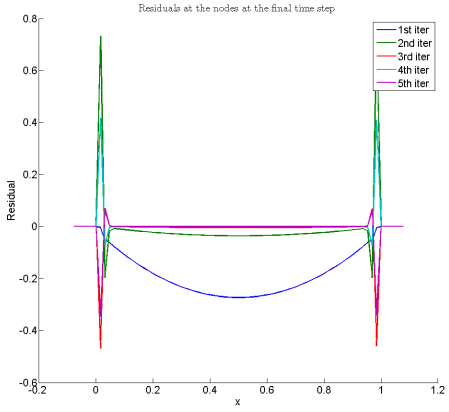
Unfortunately, this method works about as well as the Sampling method which we will see in the next section. Using the problem setup from Case 1(below) we can find the nonlinear solution



(a) The solution from $0 \leq t \leq 0.05$ with $\Delta t = 0.01$ (u_{exact} is blue)



(b) L_2 residuals values for the five time steps



(c) Residuals at first five iterations at the last time step

3 Sampling method

The other option is to use function sampling to our advantage in order to get rid ourselves of the necessity to handle three term connection coefficients. If we recall, we were able to approximate $\int f(x,t)\phi_i(x)dx$ with $f(x_i,t)$ or just the function evaluated at the nodes within our domain. Using this as an example we can see how we can approximate the integral using sampling

$$\int_0^1 \left(\sum C_j^{k,n}\phi_j(x)\right)\phi_j(x)\phi_i(x)dx = \sum C_j^{k,n}\int_0^1 \phi_j(x)\phi_i(x)dx \tag{10}$$

note that we will are using the $uk = \sum C_j^{k,n}\phi_j(x)$. If we make this assumption and then use the orthonormality properties of our scaling functions we can write a general entry of our jacobian matrix:

$$J_{i,j} = -\frac{a_1}{\Delta t}\delta_{ij} + a_2\Omega_{i,j}^{d_2} - a_3\delta_{i,j} - 2a_4u_k\delta_{i,j} \tag{11}$$

And then using the same assumptions we can write the residual vector:

$$F(u_k) = f(x,t) - u_k\left[\frac{a_1}{\Delta t} + a_3 + a_4u_k\right] + \frac{a_1}{\Delta t}u_{k-1} + a_2u_k\Omega_{i,j}^{d_2} \tag{12}$$

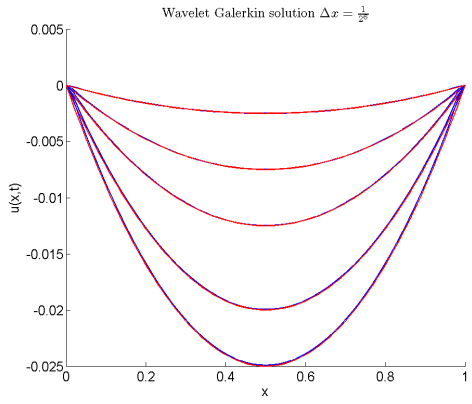
So now that we have the residual vector and jacobian matrix formulated, we must find a way to resolve the boundary conditions in a nonlinear setting. If we recall from the linear case, we extended the number of nodes in the domain past the left and right boundaries by $N-1$ in order to use the fictitious boundary approach. Also, the first and last equations in our linear system was used to enforce the boundary by adding a one in the N^{th} column and the Dirichlet boundary value in the first entry of the right hand side; the last equation was handled in a similar manner. There is another method in which truncated connection coefficients are determined near the boundary but we have decided use the fictitious boundary approach.

The problem that we have is when we apply these boundary conditions to a nonlinear iterative method. We will now present a few cases where we examine the residuals at the left boundary using a few different problem setups. Each of the setups will use the method of manufactured solutions where $u(x,t) = t(x^2-x)$ unless otherwise specified and compare a linear and a nonlinear differential equation. The linear equation will simply set $a_4 = 0$. For all of these examples we will use the $D6$ scaling function at level 6 (i.e. $\Delta x = \frac{1}{2^{level}}$).

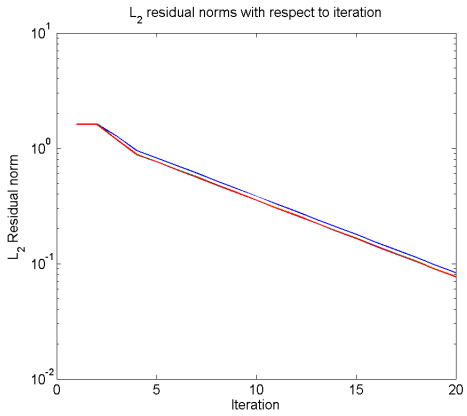
3.1 Case 1

For the first case we will keep the jacobian the same as was previously explained using the extended fictitious boundary. We will also set the N^{th} column of the first row of the jacobian to be one, but we recognize that this means we must handle the residual. Since we are solving for the change in our solution vector we want to be able to set the boundaries at the first iterate, and then not let them change throughout the iteration. Thus we will set the first N elements of the residual vector to be zero.

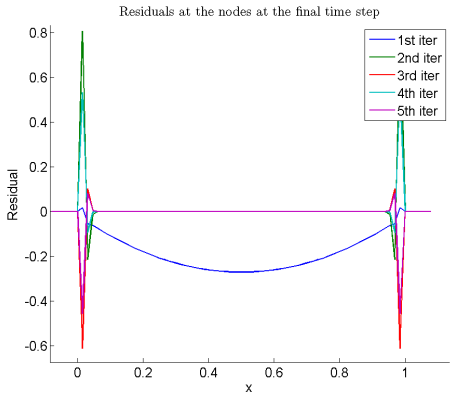
First we will look at the linear case where we set $a_4 = 0$ and adjust the right hand side accordingly:



(a) The solution from $0 \leq t \leq 0.05$ with $\Delta t = 0.01$ (u_{exact} is blue)

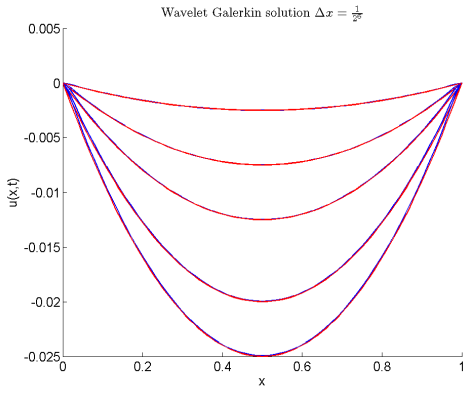


(b) L_2 residuals values for the five time steps

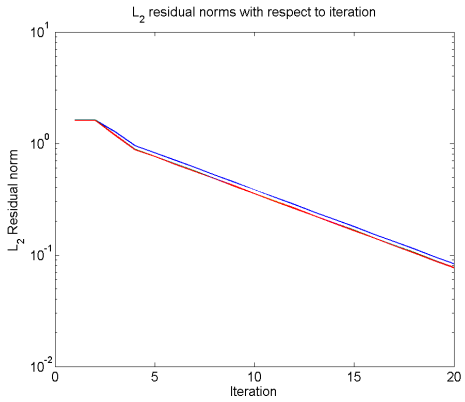


(c) Residuals at first five iterations at the last time step

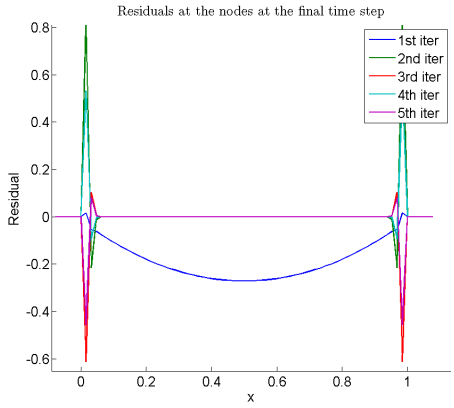
Then if we factor back in the nonlinear terms, we notice something curious:



(a) The solution from $0 \leq t \leq 0.05$ with $\Delta t = 0.01$ (u_{exact} is blue)

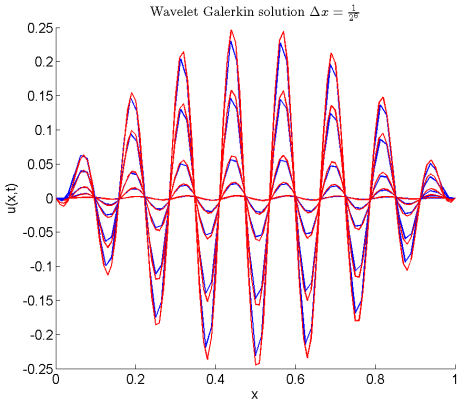


(b) L_2 residuals values for the five time steps

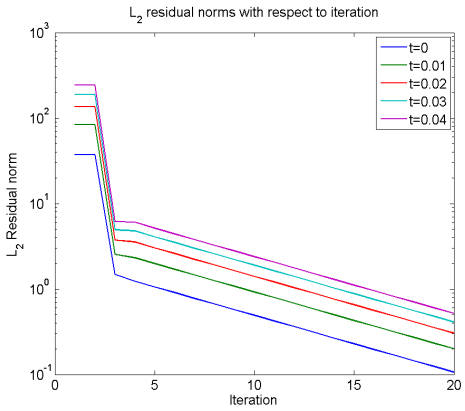


(c) Residuals at first five iterations at the last time step

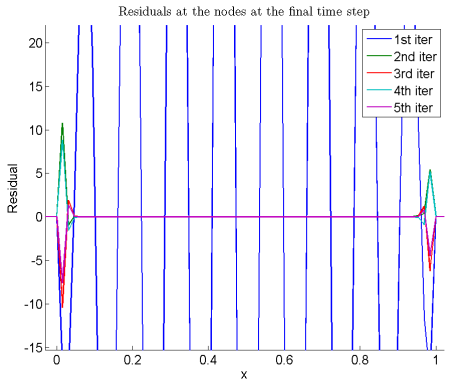
very little if anything has appeared to change. The main problem is that the boundaries take a large number of iterations to the converge regardless of whether the problem is linear or nonlinear. The interior nodes on the other hand converge effectively in a single iteration. It also appears that even though both of these methods are converging, albeit slowly, it is converging to the wrong solution as evidenced by the comparison between the exact and computed solutions. It is fair though to say that the problem we are using as an example is not truly a nonlinear problem. But if we use the example from before modified to fit a nonlinear form(see thesis) we get a similar result:



(a) The solution from $0 \leq t \leq 0.05$ with $\Delta t = 0.01$ (u_{exact} is blue)



(b) L_2 residuals values for the five time steps

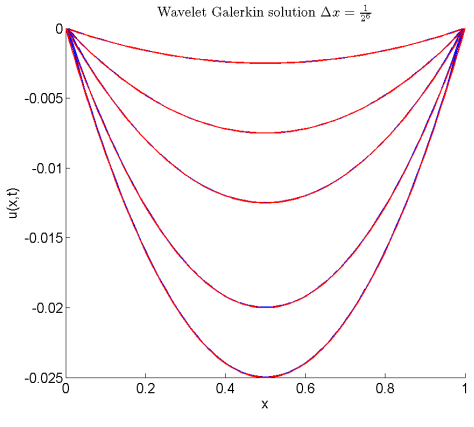


(c) Residuals at first five iterations at the last time step

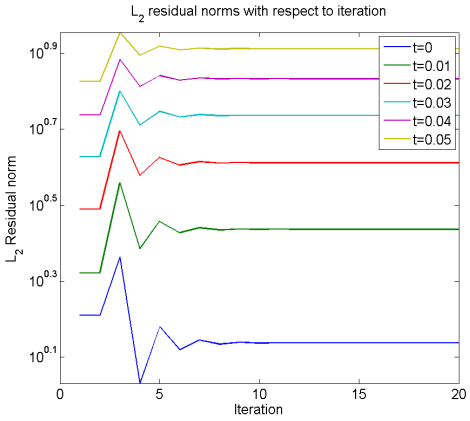
There are a few new things that we do notice from this set of plots. First in the L_2 residuals plot we notice that the residual is still decreasing at a slow rate but, as the time increases so does the L_2 norm of the residual. Also we note that the first iteration in the third plot is not in focus because we are trying to focus on the large residuals at the boundaries. The first residual is actually not very useful because it is only the iteration after our initial guess attempting to form the much more improved second guess. Unfortunately this method appears to do the best compared to the other ones we will explore.

3.2 Case 2

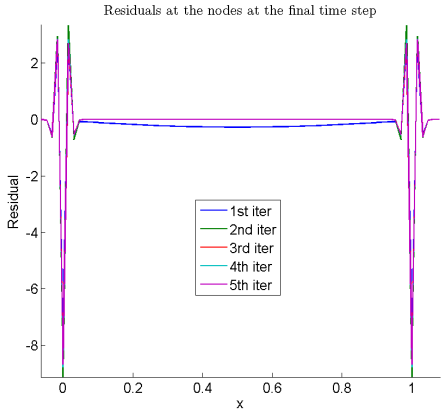
For the second case we allow only the first $N-1$ elements in our residual to be zero in an attempt to allow the scaling function defined on the boundary to play a part in the system. Computationally this really does not represent the system like the first case does. But it is the only other case that was found to even have a slight appearance of convergence. The plots for the linear case can be found below:



(a) The solution from $0 \leq t \leq 0.05$ with $\Delta t = 0.01$ (u_{exact} is blue)

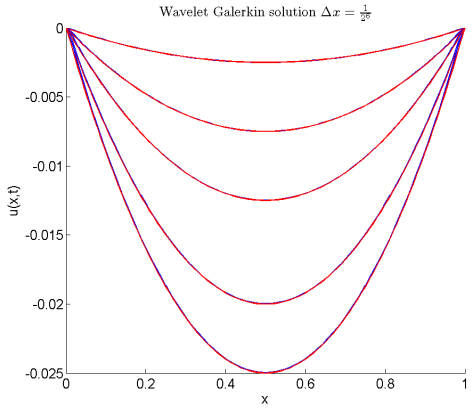


(b) L_2 residuals values for the five time steps

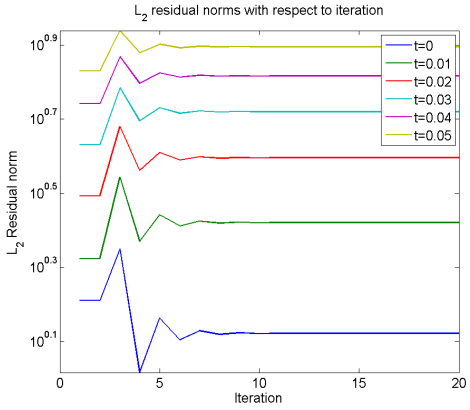


(c) Residuals at first five iterations at the last time step

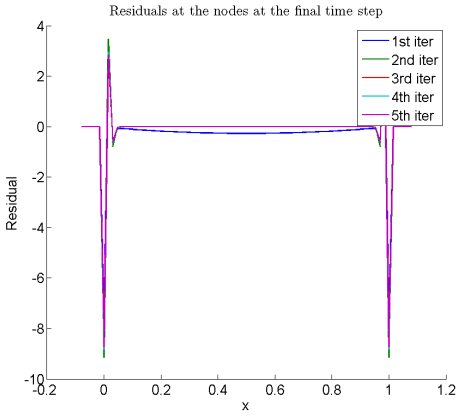
and like the first case, the nonlinear case is practically the same as the linear.



(a) The solution from $0 \leq t \leq 0.05$ with $\Delta t = 0.01$ (u_{exact} is blue)



(b) L_2 residuals values for the five time steps



(c) Residuals at first five iterations at the last time step

3.3 Other Cases

A number of other cases were tried in which the results completely blew up and failed to even appear to converge. These cases are:

1. Allow the residual to remain nonzero at any nodal point except the first element which should be used to enforce the boundary. This was tried in an attempt to allow the fictitious scaling functions the opportunity to cancel out some of the large residuals on the boundaries.

2. Use the same method as the previous item but instead extend the forcing function calculation ($f(x,t)$) past the ends of the domains. This means that the forcing function must be calculated for points that don't actually lie in our domain, not really physical.

3. Completely remove the fictitious scaling functions, making the jacobian a $Nx + 2 \times Nx$ overdetermined linear system where Nx is the number of nodes or scaling functions in our

actual domain. The added two equations were an attempt to enforce the boundary in the same manner as before.