Dirichlet boundary conditions for nonlinear systems

Brian Granzow

May 21, 2017

1 Introduction

Let $f : \mathbb{R}^N \to \mathbb{R}^N$ be a system of N nonlinear algebraic equations obtained from the finite element discretization of a system of partial differential equations. Let $x \in \mathbb{R}^N$ be the solution vector to the system

$$f(x) = 0. \tag{1}$$

Suppose we wish to solve equation (1) with Newton's method, where we iterate over the steps of solving a linear system and updating the solution vector as shown below

$$\begin{aligned} \left[\frac{\partial f}{\partial x}\right]\Bigg|_{x^{(n)}} \delta x^{(n)} &= -f(x^{(n)}) \\ x^{(n+1)} &= \delta x^{(n)} + x^{(n)}, \end{aligned} \tag{2}$$

until some convergence criterion is satisfied (e.g. $\|f(x^{(n+1)})\|_2 < \text{tol}$, where tol is a user-prescribed tolerance). Additionally, suppose we wish to impose the constraint $x_i = g_i$ for some select $i \in \{1, 2, \ldots, N\}$, corresponding to the prescription of Dirichlet boundary conditions for the discretized PDE (1). We now ask the question, how do we best impose these constraints during the process of Newton's method?

2 Albany's current implementation

2.1 Implementation details

Albany currently answers this question by enforcing the incremental solution update vector $\delta x^{(n)}$ to exactly satisfy the (known) residual at Dirichlet boundary condition degrees of freedom. That is, at each iteration n, we know we would like the updated solution

$$x_i^{(n+1)} = \delta x_i^{(n)} + x_i^{(n)}, \tag{3}$$

to exactly satisfy the constraint

$$g_i = \delta x_i^{(n)} + x_i^{(n)}, \tag{4}$$

at Dirichlet boundary condition indices i. This is achieved by requiring the increment vector δx satisfy:

$$\delta x_i^{(n)} = g_i - x_i^{(n)}. \tag{5}$$

Let $\mathcal{J} = \frac{\partial f}{\partial x}$ denote the Jacobian matrix such that $\mathcal{J} \in \mathbb{R}^{n \times n}$. Referring to the linear system in equation (2), we can impose this type of constraint on the update vector by zeroing out the appropriate row i of the Jacobian matrix and placing a one on its diagonal

$$\mathcal{J}_{ij}^{(n)} = \delta_{ij}, \tag{6}$$

(where δ_{ij} is the Kronecker delta) and by enforcing that the residual vector satisfy

$$-f_i^{(n)} = g_i - x_i^{(n)}, \tag{7}$$

or

$$f_i^{(n)} = x_i^{(n)} - g_i, \tag{8}$$

where $f^{(n)} = f(x^{(n)})$. Here we emphasize that i is *fixed* for rows corresponding to Dirichlet boundary conditions.

As a simple illustrative example, suppose we wished to fix $u_1 = g_1$ for a discretized PDE that results in a 3×3 linear system. Then for each Newton iteration, we would solve a system of the form:

$$\begin{bmatrix} 1 & 0 & 0 \\ \mathcal{J}_{21} & \mathcal{J}_{22} & \mathcal{J}_{23} \\ \mathcal{J}_{31} & \mathcal{J}_{32} & \mathcal{J}_{33} \end{bmatrix} \begin{bmatrix} \delta x_1^{(n)} \\ \delta x_2^{(n)} \\ \delta x_3^{(n)} \end{bmatrix} = - \begin{bmatrix} x_1^{(n)} - g_1 \\ R_2 \\ R_3 \end{bmatrix} \tag{9}$$

2.2 Some downsides

- The increment vector δx is, in general, not a member of the finite dimensional weighting function space $\mathcal{V}^h$ that vanishes on the domain boundaries.
- Placing a one on the diagonal introduces scaling inconsistencies if the Jacobian matrix entries are not $\mathcal{O}(1)$.
- The above can be alleviated by leaving the diagonal entry of DBC rows as is:

  $$\mathcal{J}_{ij}^{(n)} = \mathcal{J}_{ij}^{(n)} \delta_{ij} \tag{10}$$

  and scaling the residual vector as:

  $$f_i^{(n)} = \frac{g_i - x_i^{(n)}}{\mathcal{J}_{ii}} \tag{11}$$

  (no summation on i).

- Doing the above introduces a *coupling* of the computation of the residual and the Jacobian, and Albany (in general) treats these two computations as two very separate things.

3 A proposed implementation

3.1 Implementation details

Consider the imposition of Dirichlet boundary conditions to the solution vector x

$$x_i = g_i, \tag{12}$$

where we will denote this modified solution as x^g. In the context of Newton's method, the initial guess $x^{(0)}$ is modified to the vector $x^{(0)_g}$ that exactly satisfies the Dirichlet boundary conditions. We then ensure that the Dirichlet boundary condition rows never stray from their prescribed values by requiring that the corresponding rows in the Newton update vector are exactly zero

$$\delta x_i^{(n)} = 0. \tag{13}$$

This is readily achieved by zeroing out non-diagonal entries of the appropriate row in the Jacobian matrix

$$\mathcal{J}_{ij}^{(n)} = \mathcal{J}_{ij}^{(n)} \delta_{ij}, \tag{14}$$

(where δ_{ij} is the Kronecker delta, no summation on repeated indices) and specifying that the residual vector at this row is exactly zero

$$f_i^{(n)} = 0, \tag{15}$$

where $f^{(n)} = f(x^{(n)_g})$. In this instance, Newton's method will iterate over the steps

$$\begin{aligned} \left[\frac{\partial f}{\partial x}\right]\Big|_{x^{(n)_g}} \delta x^{(n)} &= -f(x^{(n)_g}) \\ x^{(n+1)_g} &= \delta x^{(n)} + x^{(n)_g}, \end{aligned} \tag{16}$$

As an illustrative example, suppose we wish to fix $u_1 = g_1$ for a discretized PDE that results in a 3×3 linear system. Then for each Newton iteration, we would solve a system of the form:

$$\begin{bmatrix} \mathcal{J}_{11} & 0 & 0 \\ \mathcal{J}_{21} & \mathcal{J}_{22} & \mathcal{J}_{23} \\ \mathcal{J}_{31} & \mathcal{J}_{32} & \mathcal{J}_{33} \end{bmatrix} \begin{bmatrix} \delta x_1^{(n)} \\ \delta x_2^{(n)} \\ \delta x_3^{(n)} \end{bmatrix} = - \begin{bmatrix} 0 \\ R_2 \\ R_3 \end{bmatrix} \tag{17}$$

3.2 Some benefits

- The increment vector δx is a member of the finite dimensional weighting function space $\mathcal{V}^h$ that vanishes on domain boundaries.
- All entries in the Jacobian matrix are now of the order of magnitude dictated by the underlying PDE without introducing a scaling and without a introducing a *coupling* of the residual and Jacobian evaluations.
- Since $\delta x_i^{(n)} = 0$ for a row i corresponding to a Dirichlet boundary condition, this row is fully *uncoupled* from all other rows and the column i can be trivially zeroed:

$$\begin{bmatrix} \mathcal{J}_{11} & 0 & 0 \\ 0 & \mathcal{J}_{22} & \mathcal{J}_{23} \\ 0 & \mathcal{J}_{32} & \mathcal{J}_{33} \end{bmatrix} \begin{bmatrix} \delta x_1^{(n)} \\ \delta x_2^{(n)} \\ \delta x_3^{(n)} \end{bmatrix} = - \begin{bmatrix} 0 \\ R_2 \\ R_3 \end{bmatrix} \tag{18}$$

3.3 Things to think about

- The proposed approach requires the pre-imposition of Dirichlet boundary conditions to the solution vector x in Albany. This solution vector is immutable (const) at all points in Albany. How should it best be modified? Likely the answer is to modify the *overlapped* solution vector. Since the 'GatherSolution' evaluator operates on this overlapped vector, this approach makes a lot of sense.
- Again the proposed approach requires the pre-imposition of Dirichlet boundary conditions to the solution vector x in Albany. I plan to implement a Phalanx 'preEvaluate' routine that performs this imposition. The downside to this is that it will be called for every single residual and Jacobian evaluation. Ideally, this pre-imposition of the solution vector would occur only at the beginning of each non-linear solve. Can this be achieved with a Nox 'PrePostOperator' or something similar?
- Since DBC rows are fully uncoupled in (18), the reduced linear system

$$\begin{bmatrix} \mathcal{J}_{22} & \mathcal{J}_{23} \\ \mathcal{J}_{32} & \mathcal{J}_{33} \end{bmatrix} \begin{bmatrix} \delta x_2^{(n)} \\ \delta x_3^{(n)} \end{bmatrix} = - \begin{bmatrix} R_2 \\ R_3 \end{bmatrix} \tag{19}$$

  could in principle be formed. I think the construction of the Tpetra maps and graphs for this reduced system would have to occur in the discretization class, as these objects are passed along to Nox/Thyra etc.. for evaluations. This would likely be a non-trivial effort.