

Flow Optimization Using Sensitivities

John Burkardt
School of Computational Science
Florida State University

07 July 2005

Contents

1	Parameterized Partial Differential Equations	3
2	The Sensitivity of a Parameterized State Variable	3
3	The Navier Stokes Equations	4
4	Newton's Method for the Continuous Navier Stokes Equations	6
5	The First Order Sensitivity Equations for R	9
6	Defining the Higher Order Sensitivity Equations for R	9
7	Computation of S, the Sensitivities	10
8	The Reduced Basis Method Derived From Taylor Expansions	11
9	Computing an Orthonormal Reduced Basis, Q	12
10	Relationships Between the Full and Reduced Spaces	12
11	Direct Solution of the Reduced Basis State Equations	13
12	The Driven Cavity	14
13	The Graded Grid Used for the Driven Cavity	14
14	Comparing Taylor Polynomials and Reduced Basis Solutions	17
15	Flow Optimization Using a Reduced Basis	22
16	Boundary and Side Conditions in the Reduced System	23
16.1	Introduction	23
16.2	The Simplest Side Conditions	23
16.3	Prediction for Conditions Dependent on the Parameter	24
16.4	Correction for Conditions Dependent on the Parameter	25
16.5	Algorithmic Implications	25

1 Parameterized Partial Differential Equations

Many physical processes can be well described by some set of *partial differential equations*. If the equations are correctly chosen, then they may not only match known information about the process, but also make correct predictions about new behavior. Thus, in order to understand a physical system, it is common to spend much effort analyzing the mathematical model instead.

In constructing a typical time independent two dimensional model of a physical process, we first identify the physical domain Ω over which the process is to be studied, with independent variables x and y , and the dependent variable or *state function* $\phi(x, y)$ whose values we are interested in.

From some physical understanding of the process, we write down the *state equations*, which describe how $\phi(x, y)$ behaves within Ω ; the *boundary conditions*, which summarize how ϕ interacts with the outside world along the boundary Γ ; and *side conditions* or other constraints to be imposed.

For this abstract introduction, we will symbolize the state equations as

$$\mathcal{L}(x, y) \phi = f(x, y) \tag{1}$$

and we will represent the boundary and side conditions as:

$$\mathcal{B}(x, y) \phi = g(x, y). \tag{2}$$

The system of state equations and boundary conditions is uniquely solvable for ϕ under certain conditions that apply to the region Ω , the differential operators \mathcal{L} and \mathcal{B} , the right hand sides f and g , and the set of functions \oplus from which we seek our answer. We assume that the requirements for solvability are always satisfied, and that there is always one and only one solution ϕ for any particular instance of Equations (1)-(2).

Now often it is not just a single problem that is of interest, but a class of problems. For instance, instead of using the particular right hand side $f(x, y)$, it might be desired to know the solution of the problems where f is replaced by $10f$, $100f$, and so on. By “and so on” we mean to suggest, of course, that we are interested in the solutions for the problem with right hand side λf , where λ is great than or equal to 1. Here λ plays the role of a *problem parameter*, that is, a scalar quantity which affects the definition of the problem to be solved.

The importance of problem parameters is so great that we are going to rewrite our previous system to explicitly include one or more parameters. Thus, our *parameterized state equation* has the form:

$$\mathcal{L}(x, y; \lambda) \phi = f(x, y; \lambda) \tag{3}$$

and our *parameterized boundary conditions* are:

$$\mathcal{B}(x, y; \lambda) \phi = g(x, y; \lambda). \tag{4}$$

Of course there are many scalar quantities which are involved in the specification of a problem. The reasons that a parameter may be singled out for explicit mention in the argument list include the fact that it represents a physical quantity which varies or can be varied experimentally, and which has some strong effect on the form of the solution.

We will now consider a means of precisely describing the strength and form of such a parametric influence.

2 The Sensitivity of a Parameterized State Variable

Let us suppose that we have a parameterized system of the form Equations (3)-(4), and that, for some particular parameter value λ_0 , we have managed to obtain a state solution. This solution may be designated as $\phi(x, y; \lambda_0)$, or, if we wish to focus purely on the parameter dependence, as $\phi(\lambda_0)$. If we suppose that the solvability and uniqueness conditions are satisfied for all λ in some neighborhood of λ_0 , then we may consider the family of solutions symbolized by $\phi(\lambda)$, for λ near λ_0 . For most physical systems, the locally-defined implicit function $\phi(\lambda)$ is continuous in λ , and may even be many-times differentiable.

Considering ϕ as a function of λ can be very useful. If the graph of the function $\phi(\lambda)$ were easily obtainable, it would help to explain the role of the parameter λ in the physical process, for instance. And if we have the ϕ and its first few derivatives with respect to λ at λ_0 , we may be able to estimate, at nearby parameter values, the value of the state function, $\phi(\lambda_0 + \Delta\lambda)$, or of some function dependent on the state function, $J(\phi(\lambda), \lambda)$.

Once we have the idea of the graph of $\phi(\lambda)$ in mind, it is clear that the most useful computational tool will be to compute the first partial derivative of ϕ with respect to λ , which we will symbolize as ϕ_λ , and which we will call the *first order sensitivity of ϕ with respect to λ* , or simply “the sensitivity”. We don’t have an explicit formula for the dependence of ϕ on λ , and so we cannot compute ϕ_λ by straightforward differentiation. Instead, we turn to the implicit relationship defined by the state system of Equations (3)-(4). When we implicitly differentiate this system, we arrive at the *sensitivity equations*:

$$\begin{aligned}\mathcal{L}_\phi\phi_\lambda &= -\mathcal{L}_\lambda\phi + f_\lambda(x, y; \lambda) \\ \mathcal{B}_\phi\phi_\lambda &= -\mathcal{B}_\lambda\phi + g_\lambda(x, y; \lambda)\end{aligned}\tag{5}$$

Note that the process of solving the sensitivity equations involves first choosing a particular parameter value λ_0 and computing the state solution $\phi_\lambda(\lambda_0)$. This defines the right hand sides; only then can the linear system of partial differential equations be solved for the sensitivities.

We will discuss *why* we want to compute solution sensitivities later. Instead, we will now move on to discuss in some detail the particular state equations that we will be subjecting to sensitivity analysis.

3 The Navier Stokes Equations

The steady incompressible flow of a viscous fluid in a two dimensional region may be completely described by three state functions: the horizontal velocity $u(x, y)$, vertical velocity $v(x, y)$, and pressure $p(x, y)$. The quantities u and v are really vector components of a single physical velocity which we may alternately write as \mathbf{u} .

Each of these functions will be called a *state function*, and a complete set of state functions will be referred to as a *state solution*, that is, enough information (say, velocities and pressure at every point in the region) which completely describes the state of the physical system under consideration.

Because they represent the behavior of a physical fluid, the functions u , v , and p obey certain physical laws. Given our assumptions about the problem, we will find it appropriate to assume that these flow functions satisfy the Navier Stokes equations for stationary incompressible viscous flow at every point (x, y) within the flow region Ω .

These equations may be written in compact vector form as:

$$-\Delta\mathbf{u} + R(\mathbf{u} \cdot \nabla\mathbf{u} + \nabla p) = 0\tag{6}$$

$$\nabla \cdot \mathbf{u} = 0\tag{7}$$

which has the equivalent scalar version:

$$-\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + R\left(u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + \frac{\partial p}{\partial x}\right) = 0\tag{8}$$

$$-\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) + R\left(u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + \frac{\partial p}{\partial y}\right) = 0\tag{9}$$

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0\tag{10}$$

Equations (8) and (9) are referred to as the *horizontal and vertical momentum equations*, while Equation (10) is known as the *continuity equation*.

The parameter R in Equations (8) and (9) is of great interest to us. Before proceeding, it may be useful to discuss its physical meaning and mathematical effect. If μ is the fluid viscosity, and ρ is the density of the fluid, then the *kinematic viscosity* is defined by:

$$\nu = \frac{\mu}{\rho} \quad (11)$$

and the parameter R is then defined by:

$$R = \frac{1}{\nu} \quad (12)$$

The parameter R in Equations (8) and (9) is known as the *Reynolds number*. In engineering practice, R is defined, in terms of “typical” or “reference” values of length, L_0 , fluid density ρ_0 , kinematic velocity U_0 , viscosity μ_0 as:

$$R \equiv \frac{\rho_0 U_0 L_0}{\mu_0} \quad (13)$$

or, if we define the *kinematic viscosity* as

$$\nu \equiv \frac{\mu}{\rho}, \quad (14)$$

then R may be defined as

$$R \equiv \frac{U_0 L_0}{\nu_0}. \quad (15)$$

Thus, it should be understood that, in Equations blah, all quantities are dimensionless; the velocities u and v have been divided by U_0 , the lengths x and y have been scaled by L_0 , the original fluid kinematic viscosity has been divided by ν_0 , and the pressures by ???.

The Reynolds number summarizes in one value the competing effects of the scales of length, velocity, mass and viscosity. In the Navier Stokes equations, R controls the relative weight of the linear and nonlinear portions of the momentum equations; as R increases, the nonlinearity grows.

Note that Equations (8) and (9) are “symmetric” in the following sense: either equation can be transformed into the other by making the substitutions $x \leftrightarrow y$ and $u \leftrightarrow v$. This means that many statements about both equations can be abbreviated to statements about just the horizontal momentum equation, with the obvious changes left to the reader.

All three Navier Stokes state equations may be symbolized by:

$$F(u, v, p) = f(x, y). \quad (16)$$

Here, $f(x, y)$ represents a source term, which will be taken to be zero throughout the region unless specifically noted.

Normally, a particular flow problem is studied over a limited or specified spatial domain, symbolized by Ω , and no information about the behavior of the flow is assumed or sought outside of Ω . However, the “outside world” exerts its influence on the flow along the boundary $\partial\Omega$. This influence may involve requirements along portions of the boundary that the flow be zero along this boundary, or have a particular normal or tangential component. There is also usually a specification of the pressure at a single boundary point. This combination of requirements is termed the *boundary conditions*; for the moment, we will represent whatever boundary conditions apply to our problem as:

$$G(u, v, p) = g(x, y). \quad (17)$$

Equation 17 is best thought of as a single equation that applies at each point $(x, y) \in \partial\Omega$. In practice, of course, the boundary conditions are likely to be expressed as a collection of equations, only some of which apply at any particular boundary point. We will ignore such complications.

It is assumed that G has no explicit dependence on R .

It will also be assumed that G is a *linear* function of (u, v, p) ; suitable boundary conditions might include $p(x_0, y_0) = 10$, $u(0, y) = y^2$, and so on. This assumption is very helpful later on. It essentially allows us

to use superposition to construct new solutions to the problem by adding solutions to the homogeneous problem.

Finally, it is assumed that if the right hand side functions f and g are replaced by zero, then the corresponding state solution is $u = v = p = 0$, regardless of the value of R . Cases in which the right hand sides of Equations (16) and (17) are everywhere zero will be referred to as the *homogenous state equation*.

4 Newton's Method for the Continuous Navier Stokes Equations

Newton's method for solving a system of nonlinear equations $H(X) = 0$ assumes that we have an approximate solution X_0 and a derivative operator $H'(X)$ which allows us to approximate the behavior of the function H around any point X_0 as a constant plus linear plus nonlinear terms:

$$H(X) = H(X_0) + H'(X_0) (X - X_0) + O(\|X - X_0\|^2). \quad (18)$$

You may be familiar with Newton's method in the case where X and the function $H(X)$ are scalar-valued. The method can be described formally in the same way for the cases where X and $H(X)$ are vectors or even functions. However, it is important in these more complicated cases to understand what kind of object H' will be, and how it can be computed.

If X and $H(X)$ are scalars, then of course $H'(X)$ is simply the usual derivative function. If X and $H(X)$ are vectors, then $H'(X)$ is the *Jacobian matrix*. But in the case of most interest to us, where we want to consider the Navier Stokes equations in their original, continuous setting, then X is a collection of functions (a vector-valued velocity function $(u, v)(x, y)$ and a scalar-valued pressure function $p(x, y)$), $H(X)$ is the Navier-Stokes equations plus boundary conditions evaluated with those functions, and H' must be some kind of linear operator. In this case, the job of H' is to estimate the value of the residual of the Navier Stokes equation at any new flow field X , based entirely on the value of the current flow field X_0 and its residual $H(X_0)$. If this is a new or unclear concept, then it is best not to worry about it now; continue with the discussion, which may help to clarify the role of H' .

As it stands, Equation (18) is true, but useless as far as computation goes! Let us simplify the equation by dropping the nonlinear terms, in which case we have the true, useless, but simpler equation:

$$H(X) \approx H(X_0) + H'(X_0) (X - X_0). \quad (19)$$

As long as $H'(X_0)$ is not singular, then in some neighborhood, this linear approximation is actually fairly good. Therefore, we might want to consider a local linear approximation $h(X)$ to $H(X)$, which we define by replacing the approximation sign by an equality:

$$h(X) = H(X_0) + H'(X_0) (X - X_0). \quad (20)$$

Equations are so much more useful than approximations! We know this model function $h(X)$ is only valid in a small neighborhood of X_0 , but since $h(X)$ is a linear function, we can analyze it completely. One of the most interesting things about a function is to find where it is zero. A formula for the root X^* for which $h(X^*) = 0$ is instantly obtainable:

$$x^* = X_0 - (H'(X_0))^{-1} H(X_0). \quad (21)$$

Of course, this is just a root of $h(X)$; but if $h(X)$ is a good model of $H(X)$, then it is reasonable to assume that x^* is a good estimate of the root X^* , which we presume exists and is nearby.

We are actually going to be generating a sequence of estimates for the root. It is common to index such an iterative sequence with a superscript. Thus, we regard the point X_0 as our initial estimate of the root, and now we call it X^0 . The value that we got from Equation (21) would then be indicated by X^1 . If X^1 is a better approximation to the root than X^0 was, then it is presumably a good idea to reconstruct the model linear function base at X_1 and solve for the root of this new linear equation. But that simply means that on our next step, we want to evaluate the right hand side of Equation (21) at X^1 now.

In many cases, this procedure, *the Newton iteration*, will produce a sequence of rapidly improving estimates for the root of the original nonlinear function. The Newton iteration consists of repeatedly evaluating the right hand side of Equation (21) and using the result as the new estimate of the root:

$$X^{k+1} = X^k - (H'(X^k))^{-1} H(X^k). \quad (22)$$

Note that we can also write this equation in the equivalent implicit form:

$$H'(X^k) (X^{k+1} - X^k) = -H(X^k). \quad (23)$$

in which case, we have to solve some sort of linear algebraic equation or linear functional equation, for the *Newton increment*

$$\Delta X^k \equiv X^{k+1} - X^k. \quad (24)$$

Depending on the situation, we may prefer the explicit or implicit form of the Newton iteration, and we may want to talk in terms of the next iterate X^{k+1} or the increment ΔX^k .

To apply Newton's method to our flow problem, we need a way of defining the derivative operator H' of the function H . This is a simple task in the typical discretized case, when both H and its argument X are vectors of real numbers. But we need to be able to define H' in the continuous case as well, where H and X are functions, or even sets of functions (horizontal and vertical velocity and the pressure, for instance). In such a case, the simple vector calculus definitions of the derivative as a Jacobian matrix are not useful.

To see what is going on, it really helps to go back to the definition of the derivative as the limit of difference quotients. Let us start with any evaluation point X_0 . We pick an arbitrary increment function ΔX . We now want to compare the value of H at X_0 to its values along the one dimensional family of points defined by $X_0 + \epsilon \Delta X$.

This allows us to define a difference quotient whose divisor is a real number. If the resulting expression has a limit, then that limit is termed the *Gateaux derivative* of H in the direction ΔX :

$$H'_G(X_0)(\Delta(X)) \equiv \lim_{\epsilon \rightarrow 0} \frac{H(X_0 + \epsilon \Delta X) - H(X_0)}{\epsilon}. \quad (25)$$

Of course, from the definition, the Gateaux derivative only gives information along a particular direction; if the Gateaux derivative exists at X_0 for a particular increment direction ΔX , then it may or may not exist for another increment direction; and if it exists for two distinct increment directions from X_0 , there is no guarantee that the two Gateaux derivatives are equal! However, the definition of the Gateaux derivative should start to give you a picture of what it might mean when we speak of the derivative of a function of a function.

In most interesting cases, the Gateaux derivative exists, and is independent of the increment direction, and is continuous in some neighborhood of the point of interest X_0 . In this case, it graduates to being a *Frechet derivative*. Although Frechet derivatives are really just the same kind of difference quotient as Gateaux derivatives, their description might seem different, since they are usually described in terms of differences between the value of H at X_0 and a second nearby point X_1 :

$$\lim_{\|X_1 - X_0\| \rightarrow 0} \frac{\|H(X_1) - H(X_0) - H'_F(X_0)(X_1 - X_0)\|}{\|X_1 - X_0\|} = 0. \quad (26)$$

For our work, the typical argument X_0 will be the triple of state functions (u, v, p) ; the nonlinear function H will be the Navier Stokes equations, which we will write $F = f$, and the boundary conditions, which we will write $G = g$. We will assume that the function H is always Frechet differentiable. Newton's method requires a "suitable" starting estimate $X = (u, v, p)$ of the solution, which can be any functions of (x, y) defined over Ω , and which need not satisfy the boundary conditions.

We may then symbolize the unknowns as:

$$X \equiv \begin{pmatrix} u \\ v \\ p \end{pmatrix} \quad (27)$$

and the system of nonlinear equations as:

$$H(X) \equiv \begin{pmatrix} F(u, v, p) - f \\ G(u, v, p) - g \end{pmatrix}. \quad (28)$$

Our search for a solution of the original problem can be represented as seeking functions X so that:

$$H(X) = 0. \quad (29)$$

We may compute the derivative operator for this nonlinear function by carrying out the limit operation. We start with an initial estimate X^0 for the solution. We set up the implicit form of Newton's method: the operator H' is evaluated at X^0 , and applied to the unknown solution increment ΔX^0 , with the result set to zero:

$$H'(X^0) (\Delta X^0) = 0. \quad (30)$$

Solving this equation, we then get our first Newton iterate:

$$X^1 = X^0 + \Delta X^0. \quad (31)$$

For the Navier Stokes problem, this equation, when written out, represents the linearization of the operators $F - f$ and $G - g$, evaluated at the solution (u_0, v_0, p_0) and operating on the increment $(\Delta u, \Delta v, \Delta p)$.

To work out the linearization operator $H'(X_0)(\Delta X)$ for the Navier Stokes equations are, you should simply carry out the usual difference quotient estimation using a Gateaux derivative approach. We will write $X_0 = (u_0, v_0, p_0)$ for the flow values, and $\Delta X = (\Delta u, \Delta v, \Delta p)$ for the increment direction. Now, write down the Navier Stokes equations for the functions (u_0, v_0, p_0) , and then write them down again for $(u_0 + \epsilon \Delta u, v_0 + \epsilon \Delta v, p_0 + \epsilon \Delta p)$. (You can probably save some time if you assume you know that will happen to the simple linear terms in the equations.) Subtract the two equations divide by ϵ , and then let ϵ go to zero to get the value of $H'(X_0)(\Delta X)$.

The linearized Navier Stokes equations you will arrive are known as the *Oseen equations*. These are a sort of set of flow equations for the increment functions. Note, in particular, that the velocity increment functions satisfy the same continuity equations as the velocity functions do:

$$-\left(\frac{\partial^2 \Delta u}{\partial x^2} + \frac{\partial^2 \Delta u}{\partial y^2}\right) + R \left(\Delta u \frac{\partial u_0}{\partial x} + u_0 \frac{\partial \Delta u}{\partial x} + \Delta v \frac{\partial u_0}{\partial y} + v_0 \frac{\partial \Delta u}{\partial y} + \frac{\partial \Delta p}{\partial x}\right) = 0 \quad (32)$$

$$-\left(\frac{\partial^2 \Delta v}{\partial x^2} + \frac{\partial^2 \Delta v}{\partial y^2}\right) + R \left(\Delta u \frac{\partial v_0}{\partial x} + u_0 \frac{\partial \Delta v}{\partial x} + \Delta v \frac{\partial v_0}{\partial y} + v_0 \frac{\partial \Delta v}{\partial y} + \frac{\partial \Delta p}{\partial y}\right) = 0 \quad (33)$$

$$\frac{\partial \Delta u}{\partial x} + \frac{\partial \Delta v}{\partial y} = 0 \quad (34)$$

You should study these equations, and notice the similarities and differences compared to the Navier Stokes equations. Keep in mind the distinction between the flow quantities and the increment quantities. The flow field quantities are *fixed, known, given functions*; the increment quantities are *freely varying, small, unknown increment functions*; they represent small changes to an existing flow.

Of course, for a given problem, we also have to differentiate whatever side conditions and boundary conditions we have imposed. Formally, the linearized boundary conditions will be part of our operator H' , and will have the following form:

$$G'(u_0, v_0, p_0) (\Delta u, \Delta v, \Delta p) = G_u(u_0, v_0, p_0)\Delta u + G_v(u_0, v_0, p_0)\Delta v + G_p(u_0, v_0, p_0)\Delta p = 0. \quad (35)$$

We have had to leave the boundary conditions in this fairly general form, since we wish to be free to choose a variety of different conditions. Usually, the actual boundary equations being employed will be so simple that their linearization will be no challenge at all.

In fact, for our current problem, recall that we assumed the boundary conditions were linear in the unknowns. This implies that the linearized boundary conditions are formally identical to the original boundary conditions on the left hand side, but with a homogeneous right hand side.

For instance, an original boundary condition $u(x, 0) = x^2$ would, after differentiation and evaluation at the increment $(\Delta u, \Delta v, \Delta p)$, become $\Delta u(x, 0) = 0$. The fact that the boundary conditions for the linearized problem don't change much from those for the nonlinear problem will be helpful later.

5 The First Order Sensitivity Equations for R

To derive the first order sensitivity equations, we begin by assuming that we have a particular solution (u^0, v^0, p^0) of the Navier Stokes equations (8)-(10), with suitable boundary conditions. We are interested in the first derivatives of the solution components with respect to the parameter R . To find equations that specify the behavior of these functions, we simply differentiate the Navier Stokes equations and boundary conditions with respect to R , and interchange differentiations wherever possible so that differentiation with respect to the parameter is performed first. We now introduce the shorthand:

$$u^1 \equiv \frac{\partial u}{\partial R} \quad (36)$$

which we promise will not result in any ambiguity, because no powers of u , v or p will be used.

We have already discussed the fact that the differentiated boundary conditions will have the same form as the original boundary conditions, but with a homogeneous right hand side. This will remain true, no matter how many times we repeat the differentiation process. This means we can drop consideration of the boundary conditions for a while, and turn to the more interesting question of what happens to the state equations under repeated differentiation.

Using our shorthand, the resulting differentiated Navier Stokes equations are:

$$\begin{aligned} -\left(\frac{\partial^2 u^1}{\partial x^2} + \frac{\partial^2 u^1}{\partial y^2}\right) + R \left(u^1 \frac{\partial u^0}{\partial x} + u^0 \frac{\partial u^1}{\partial x} + v^1 \frac{\partial u^0}{\partial y} + v^0 \frac{\partial u^1}{\partial y} + \frac{\partial p^1}{\partial x}\right) \\ = -(u^0 \frac{\partial u^0}{\partial x} + v^0 \frac{\partial u^0}{\partial y} + \frac{\partial p^0}{\partial x}) \end{aligned} \quad (37)$$

$$\begin{aligned} -\left(\frac{\partial^2 v^1}{\partial x^2} + \frac{\partial^2 v^1}{\partial y^2}\right) + R \left(u^1 \frac{\partial v^0}{\partial x} + u^0 \frac{\partial v^1}{\partial x} + v^1 \frac{\partial v^0}{\partial y} + v^0 \frac{\partial v^1}{\partial y} + \frac{\partial p^1}{\partial y}\right) \\ = -(u^0 \frac{\partial v^0}{\partial x} + v^0 \frac{\partial v^0}{\partial y} + \frac{\partial p^0}{\partial y}) \end{aligned} \quad (38)$$

$$\frac{\partial u^1}{\partial x} + \frac{\partial v^1}{\partial y} = 0 \quad (39)$$

An important thing to notice is that, once we have moved the known quantities to the right hand side, the left hand side is identical in form to that of the Newton increment equations.

6 Defining the Higher Order Sensitivity Equations for R

If we are interested in computing higher order sensitivities, then we must take the first order sensitivity equations and differentiate them with respect to R repeatedly. This process rapidly becomes tedious, and is a major defect of the reduced basis method. However, in this particular setting, we can easily deduce the form of *all* higher order sensitivity equations.

First, we adhere to the convention that the known terms are moved to the right hand side. Then the left hand side of the equations always has the same form, except that the unknowns are of one higher order than for the previous order equation.

Secondly, we note that the continuity equation will never have a nonzero right hand side, and so we need consider it no further.

Thirdly, the fact that the vertical momentum equation is transformed into the horizontal momentum equation by the simple interchange discussed earlier means that we can also forget about the vertical momentum equation, leaving as our only unknown the form of the right hand side of the horizontal momentum equation.

After some reflection, it can be seen that the right hand side of the horizontal momentum equation of order n will have the following form:

$$\begin{aligned}
 RHS_n &= -R \left(\sum_{i=1}^{n-1} \binom{n}{i} (u^{n-i} u_x^i + v^{n-i} u_y^i) \right) \\
 &\quad - \left(\sum_{i=0}^{n-1} \binom{n-1}{i} (u^{n-i-1} u_x^i + v^{n-i-1} u_y^i) \right)
 \end{aligned} \tag{40}$$

A single computer procedure can evaluate this formula for any value of n , always assuming the availability of the values of the lower order sensitivities. Thus, the computation of the sensitivities can easily be automated, avoiding the drudgery and overhead of hand or symbolic differentiation.

7 Computation of S, the Sensitivities

We assume that our large scale computation requires the values of the state variables at one or more sets of parameter values. We will generally assume that the state variables are found using Newton's method, or the generalized Newton method that does not re-evaluate the Jacobian matrix on each iterative step. In the latter case, it is assumed that the Jacobian does not change significantly enough to affect the results.

The sensitivities are to be computed as part of a fluid flow calculation. Once the Newton iteration has converged, and has produced a state solution, we will compute the sensitivities of that state solution. To do so, we need only solve a linear system, whose system matrix is the same as that employed by Newton's iteration. Strictly speaking, this matrix should be re-evaluated at the final state solution in order to be up to date. However, for efficiency, we will use the Jacobian as evaluated at the previous Newton iterate, or indeed, several iterates back, in the case of the generalized Newton method.

We accept these approximations because it is generally extremely expensive to evaluate and factor the Jacobian matrix. If we make do with the currently factored matrix, then each sensitivity vector can be computed for the price of a linear back-solve, which is cheaper than a factorization by roughly an order of magnitude in the number of equations.

Note that each sensitivity equation requires the values of all lower order sensitivities in order to evaluate the right hand side. Thus, even if only the k -th sensitivity is actually needed, all lower orders must be computed. Moreover, the sensitivities are defined recursively, so it is not possible to set up all k right hand sides at once; rather, repeated cycles must be carried out, involving setting up the right hand side of a given order, back-solving for the sensitivity of that order, and then moving up to the next order.

There are two remaining difficulties in the computation of sensitivities.

First, it is necessary to derive the form of the right hand side for each desired sensitivity. This process can be tedious, and the resulting expressions tend to grow in length as the order increases. In some cases, it is possible to determine a simple recursion formula that allows for a more efficient generation of these right hand sides. We are fortunate in that our parameter occurs explicitly. In cases where the parameter is implicit, such as giving the location of a boundary, the differentiation is much more difficult.

Secondly, it is necessary for the program to be able to determine the quantities that show up in the right hand side. This is not a problem when we are considering the REYNLD parameter, but can be very harmful when the parameter is implicit. If, for instance, geometric data depends on the parameter, then we generate higher order spatial derivatives which our finite element program approximates with lower accuracy, thus degrading our results.

8 The Reduced Basis Method Derived From Taylor Expansions

If the state variables (u, v, p) are smoothly differentiable in x, y , and R , then it is legitimate to interchange the orders of differentiation, which was used to derive the equations for the higher order sensitivities, and the results are indeed the usual derivatives of the state variables with respect to the parameter. Such derivatives provide extremely useful information about the “local” behavior of the state variables; that is, how they are affected by small changes in R .

In particular, if we assume that the state variables (u, v, p) are sufficiently differentiable, and if we have a state solution (u^0, v^0, p^0) for some parameter value $R = R^0$, then for “small” perturbations ΔR and for any nonnegative integer N , the N -th order Taylor expansion holds for u (and for the other state variables):

$$u(R^0 + \Delta R) = u^0 + \sum_{i=1}^N \frac{\Delta R^i}{i!} u^i + O((\Delta R)^{N+1}), \quad (41)$$

where the notation u^i is a useful shorthand term:

$$u^i \equiv \frac{\partial^i u}{\partial R^i}(R^0). \quad (42)$$

If N is large enough, and the perturbations of R are small, the error term in the Taylor expansion (41) may be dropped, and the remaining right hand side used to approximate values of $u(R^0 + \Delta R)$ near the known value $u^0 \equiv u(R^0)$.

However, the Taylor approximation explicitly warns us that, for a fixed N , our approximation error is likely to grow like a polynomial of order N as we move away from the base solution. If it is desired to approximate u over a large range of perturbations, then one solution would be to increase N as necessary. However, it is also possible to proceed by changing the definition of the coefficients of the u^i in Equation (41), while keeping N fixed.

A justification for this approach requires arguing that the first few derivatives of the flow exhibit a series of dominant modes of behavior for the problem, which persist over a wide range of parameter values, even though the actual weights of these modes might not correspond to those prescribed by the Taylor series. To say this is to assert that, for a fixed set of N_{rb} basis vectors and some now *unknown* coefficients c , it should still be true that:

$$u(R^0 + \Delta R) \approx u^0 + \sum_{i=1}^{N_{rb}} c_i (\Delta R) u^i. \quad (43)$$

over a “wide” range of values of ΔR .

REFUGEE EQUATION:

$$(u, v, p)(R) \approx (u^0, v^0, p^0) + \sum_{i=1}^{N_{rb}} (u^i, v^i, p^i) c_i. \quad (44)$$

Now if we restrict ourselves to considering only solutions lying in the affine space with origin u^0 , of linear combinations of the vectors u^i for $i = 1 \dots N_{rb}$, then we see that the quantities c_i play the role of coordinates, and the u^i play the role of coordinate axes. Since the u^i generate this linear subspace of approximate solutions, we refer to them as a *reduced basis* for that space. Naturally, other sets of vectors may generate the same space; if the u^i are linearly dependent, or nearly so, a smaller number of vectors may suffice. And it will shortly be to our advantage to replace the non-orthonormal vectors by an orthonormalized set.

Note that a coefficient c_0 was *not* included, to multiply the term u^0 . Such a coefficient would not produce proper solutions for the current problem; the base solution u^0 satisfies the non-homogenous boundary conditions $G(u, v, p) = g(x, y)$ exactly. A multiple of u^0 would not satisfy them. As we will see, in most cases, the sensitivities will satisfy homogeneous boundary conditions, so the coefficient c_0 *must* be 1 for our problems, and so we can drop it (implicitly setting it to 1). Under the given assumptions about the boundary

conditions, and with c_0 implicitly set to 1, *any* combination of coefficients c_i , $i = 1, N_{rb}$ will automatically satisfy the boundary conditions imposed by Equation (17).

The method we have described here is known as the *Taylor reduced basis method*, since, in analogy to the Taylor expansion, all the sensitivities are evaluated at the base solution.

A different method is known as the *Legendre reduced basis method*, which computes a base solution at N_{rb} different values of the parameter, and derives suitable expansion functions in terms of differences of these base solutions. For details, see Gunzburger [4].

9 Computing an Orthonormal Reduced Basis, Q

We have computed a collection of vectors S , the first N_{rb} derivatives of the solution with respect to the parameter. We have assumed that these vectors contain the predominant modes of change of the solution, and we would like to use them as a basis for approximating the set of solutions near the given value of R .

But while each column of S has an obvious meaning, the collection of derivative vectors is not immediately suitable for use as a basis. We must first *orthonormalize* the set, for accuracy and efficiency. This can be done by applying the QR factorization of S , as provided, for instance, by the LAPACK subroutine DGEQRF [1], so that:

$$S = Q R, \tag{45}$$

where Q is an N by N_{rb} matrix with orthogonal columns, and R is an N_{rb} by N_{rb} upper triangular matrix.

For our purposes, R is of little further interest, except that the size of the diagonal entries R_{ii} can tell us if the i -th reduced basis vector is nearly linearly dependent on its predecessor, in which case we should drop it.

Q , however, is extremely useful, since its columns span the same space as S , while being orthonormal. Therefore, we will consider the equivalent approximation based on Q . We will again use c to stand for the vector of as unknown coefficients, but they will now be coefficients for the columns of Q , rather than S . Any given state solution $(u, v, p)(R)$ will be approximated by:

$$(u, v, p)(R) \approx (u^0, v^0, p^0) + Q c. \tag{46}$$

10 Relationships Between the Full and Reduced Spaces

We now consider the relationships between the full and reduced spaces, and, in particular, how they may be carried out via the orthonormal transformation matrix Q .

First, note that u^0 lies in the space $Whoopi^N$ of solutions to the state equation with the given boundary conditions, and for $i > 0$, u^i lies in the the space $Whoopi_0^N$ of solutions to the same state equations with homogeneous boundary conditions. Therefore, since every element of the affine space *Goldberg* is the sum of u^0 and an arbitrary combination of elements of $Whoopi_0^N$, such elements all satisfy the nonhomogenous boundary conditions.

Because the columns of Q are orthonormal, we have:

$$Q^T Q = I(N_{rb}), \tag{47}$$

where $I(N_{rb})$ is the N_{rb} by N_{rb} identity matrix. This means that, if we start in the reduced space with a set of coefficients c_{rb} , and use Q to expand c_{rb} to a set of coefficients c_{fl} in the full space, and then apply Q^T to c_{fl} to return to the small space, the result is our starting solution c_{rb} , or in other words:

$$c_{rb} \xrightarrow{Q} c_{fl} \xrightarrow{Q^T} c_{rb} \tag{48}$$

However, the *rows* of Q are not orthonormal, and so in general:

$$Q Q^T \neq I(N). \tag{49}$$

which implies that if we start with a set of coefficients in the full space, c_{fl} , use Q^T to transform to c_{rb} in the reduced space, and then apply Q to compute c'_{fl} , then in general,

$$c_{fl} \xrightarrow{Q^T} c_{rb} \xrightarrow{Q} c'_{fl} \neq c_{fl}. \quad (50)$$

Instead, the operator QQ^T is a *projection* of the full solution space onto the affine space.

Thus the matrix Q , as defined in Equation (45), embodies the relationship between the full space and the reduced space, projecting (reducing) or expanding solutions from one space to the other. This makes it easy to consider the restriction of the domain of our finite element function to the image of the reduced space, and then to redefine that function to have the reduced space as its domain:

$$H(Q c_{rb}) \quad (51)$$

Now if we were to try to solve the system of equations

$$H(Q c_{rb}) = 0, \quad (52)$$

then we would have difficulties, since we have N equations in N_{rb} variables. It might then occur to us to use the matrix Q^T as a premultiplier of the function H , defining a new function H_{rb} as:

$$H_{rb}(c_{rb}) \equiv Q^T H(Q c_{rb}) \quad (53)$$

Moreover, if we compute the jacobian of this reduced system, we arrive at

$$H'_{rb}(c_{rb}) = Q^T H'(Q c_{rb}) Q \quad (54)$$

Note that H'_{rb} cannot be singular unless H' is as well. That is, suppose that

$$H'_{rb}(c_{rb}) x_{rb} = 0 \quad (55)$$

where H'_{rb} has been evaluated at some point c_{rb} , and multiplies some nonzero vector x_{rb} . Then we can evaluate H at the point $Q c_{rb}$, and apply it to the vector $Q x_{rb}$, (which is nonzero since x_{rb} is nonzero and $Q^T Q$ is the identity), to arrive at:

$$0 = H'_{rb}(c_{rb}) x_{rb} = Q^T H'(Q c_{rb}) Q x_{rb} \quad (56)$$

and if we multiply on the left by x_{rb}^T , we arrive at OOPS, I DON'T KNOW WHAT I'M DOING HERE!!

Once we see how Q can help us, it is natural to seek a way of *justifying* this use. We began by regarding the columns of Q as defining a *reduced basis* for the original finite element space; we may instead choose to regard Q as the *basis* for a smaller, but more natural, finite element space.

11 Direct Solution of the Reduced Basis State Equations

Let us now consider the question of searching for solutions of the discretized state equations $G^h(u, v, p)$ which are restricted to lie exactly within the affine space passing through the fundamental solution (u^0, v^0, p^0) , and spanned by the first N_{rb} derivative vectors u^i . Our solution space was originally restricted to the N -dimensional space of sums of the finite element basis functions, and we are now further restricting our solution space, so that our solution has the form:

$$H^h((u^0, v^0, p^0) + Q c_{rb}) = 0. \quad (57)$$

However, this represents a set of N nonlinear equations in N_{rb} unknowns, where N is generally much larger than N_{rb} . It is natural to define the *reduced basis residual* as:

$$H_{rb}(c_{rb}) \equiv Q^T H^h((u^0, v^0, p^0) + Q c_{rb}), \quad (58)$$

which defines a set of N_{rb} nonlinear functions in N_{rb} unknowns. Moreover, since the Jacobian of H_{rb} is

$$H'_{rb}(c) = Q^T H'((u^0, v^0, p^0) + Q c_{rb})Q, \quad (59)$$

the system

$$H_{rb}(c_{rb}) = 0 \quad (60)$$

is solvable whenever the matrix Q has full row rank, and the full system $H((u^0, v^0, p^0) + Q c_{rb}) = 0$ is solvable.

It is important to see that there is a second way of arriving at this same system of equations. Suppose, for whatever reasons, we take the columns of the matrix Q to be the basis vectors of a finite element formulation of the discretized problem. Then the resulting finite element equations would have exactly the form of Equation (60). This fortunate fact allows us to appeal to many of the results of finite element analysis for our reduced basis method.

Moreover, because we were careful to orthogonalize the matrix Q , we have a simple way of characterizing the relationship between the space of solutions defined by the N_{rb} reduced coefficients c and the previous N -dimensional finite element space. In particular, any set of reduced coefficients c_{rb} defines a corresponding set of full coefficients c_{fl} by the injection:

$$c_{fl} \equiv c_{fl0} + Q c_{rb} \quad (61)$$

while any set of full coefficients can be projected into the space of reduced coefficients by the projection:

$$c_{rb} \equiv Q^T (c_{fl} - c_{fl0}). \quad (62)$$

12 The Driven Cavity

It is now time to consider a model problem, to which we will apply the techniques discussed so far. The problem we choose is known as *the driven cavity*, and has been widely studied and discussed. In particular, Peterson [8] studied the behavior of the reduced basis method applied to this problem. The simplest version of the problem is set in two dimensions. A viscous fluid is contained in a square region with sides of length 1. The sides and bottom of the region are no-slip walls, that is, the fluid immediately in contact with the walls is motionless, so that $u = v = 0$. Along the top of the region, some force impels the fluid to move from left to right with a given speed U . The fluid viscosity ν transmits this force throughout the cavity, causing the fluid to rotate clockwise in a simple vortex. If the impelling velocity U is increased, the fluid rotates faster, but also begins to assume more complicated forms, spawning smaller subvortices. The Reynolds number for this flow is

$$R = \frac{\text{velocity} * \text{length}}{\text{dynamic viscosity}} = \frac{U \cdot 1}{\nu} \quad (63)$$

13 The Graded Grid Used for the Driven Cavity

As the Reynolds number increases, the solution of the steady Navier Stokes problem becomes more difficult. This problem shows up in the divergence of the Newton iteration; as R increases, the ball of attraction decreases in size, and eventually it is infeasible to proceed.

One remedy for this problem is simply to uniformly reduce the size of the elements; this will permit the computation to proceed to somewhat higher values of R before the breakdown problem recurs. However, such mesh refinement quickly makes the problem much too large to solve, and so it is important to try to get the best results out of a given size mesh as possible.

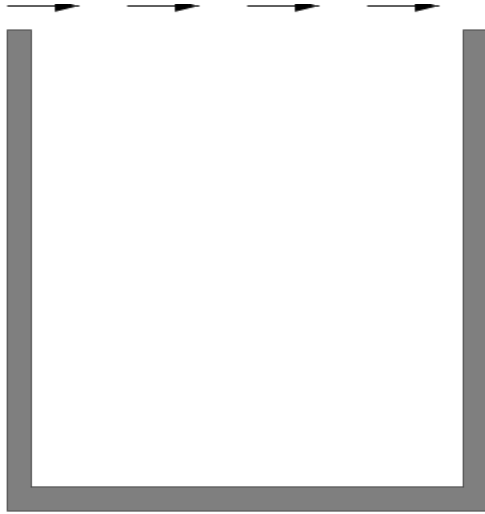


Figure 1: The geometry of the driven cavity.

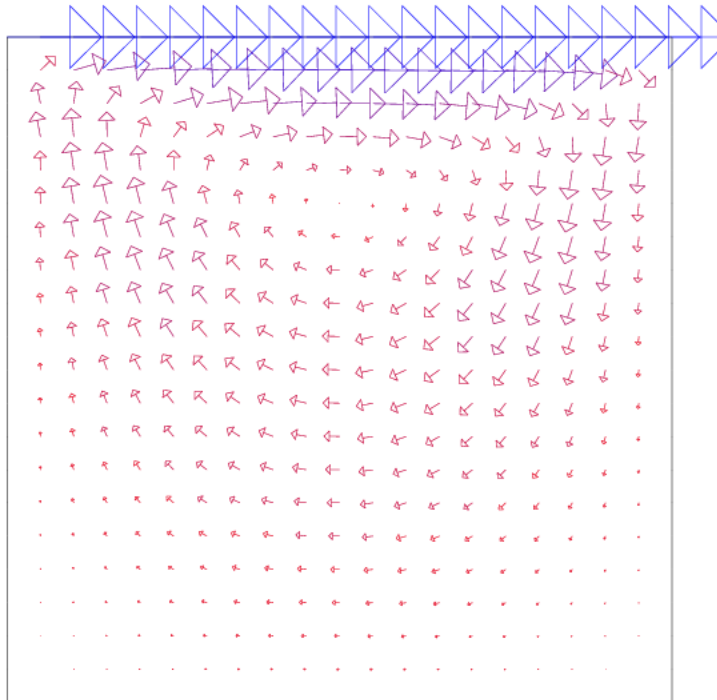


Figure 2: The computed velocity for a driven cavity with $V=10$, $Re=1$.

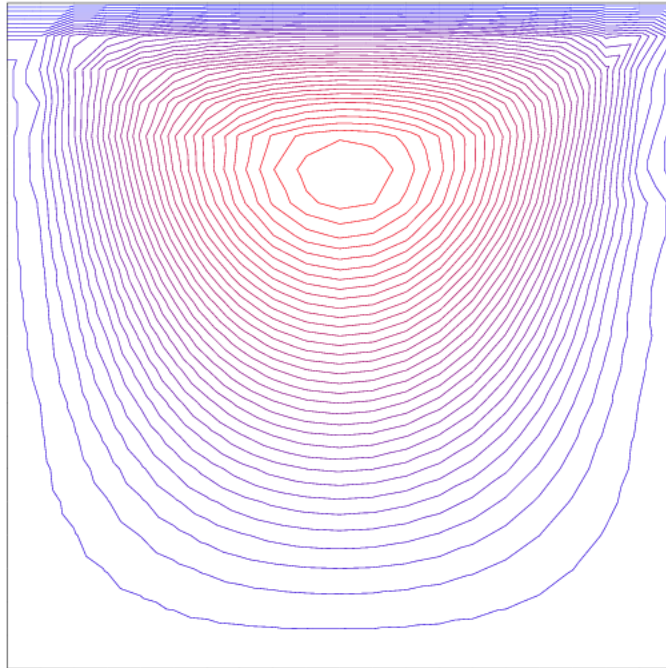


Figure 3: The stream lines for a driven cavity with $V=10$, $RE=1$.

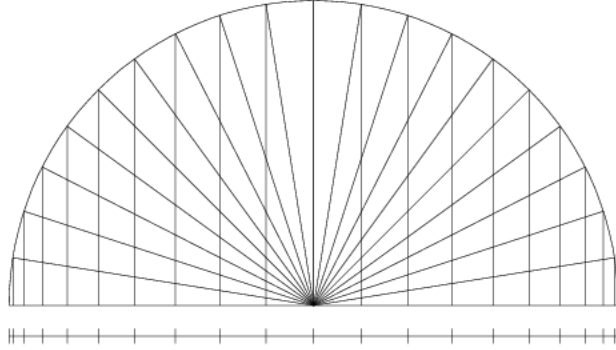


Figure 4: Using the cos function to generate a 1-d mesh.

In particular, for the driven cavity, there are better meshes to use than the uniform one. One scheme for producing mesh points uses the cos function to cluster mesh points near the boundaries, using the formula:

$$\theta_i = \frac{(n-i)\pi}{(n-1)} \text{ for } i = 1, \dots, n \quad (64)$$

$$x_i = \frac{(1 - \cos(\theta_i))x_{left} + (1 + \cos(\theta_i))x_{right}}{2} \quad (65)$$

where the dimension x ranges from x_{left} to x_{right} . This formula is equivalent to constructing a semicircle of radius $(x_{left} - x_{right})/2$ on the segment $[x_{left}, x_{right}]$, drawing n equally spaced rays within the semicircle, and dropping perpendiculars from each ray to the segment, as suggested in Figure (4). The same formula may be used to generate the y coordinates of mesh points. The resulting 2-d mesh, for the case $n = 21$ is shown in Figure (5).

A heuristic explanation for why this mesh is better depends on the fact that the Navier Stokes equations can be thought of as describing the behavior of two forces: a linear diffusion force that dominates at low Reynolds numbers and away from walls, and a nonlinear force that is strong near walls or high velocity gradients, and for high Reynolds numbers. As the Reynolds number increases, a grid should begin to cluster more points near walls, where there is a large spatial velocity gradient.

14 Comparing Taylor Polynomials and Reduced Basis Solutions

Let us suppose that we have a parameter value R_0 , a (full basis) flow solution $u(R_0)$, and the first n partial derivatives of u with respect to R_0 , namely $u'(R_0), \dots, u^n(R_0)$. Then for small perturbations ΔR , we may use the n term Taylor polynomial to approximate the flow solution $u(R_0 + \Delta R)$.

We can regard the finite Taylor expansion as constructing an approximation to the solution at a “nearby” parameter value by specifying both an affine space $\{u(R_0)\} \oplus T^n(u(R_0))$ and the coordinates of a particular element of that space. Notice that the reduced basis method, with n vectors, uses the same affine space, but differs in its choice of a particular approximating candidate from that space.

The Taylor approximation is familiar, and its approximating behavior easy to understand; thus it provides a useful comparison with the reduced basis approximation. The Taylor approximation for a dependent variable u at a base parameter value R_0 of order n includes the following typical behaviors:

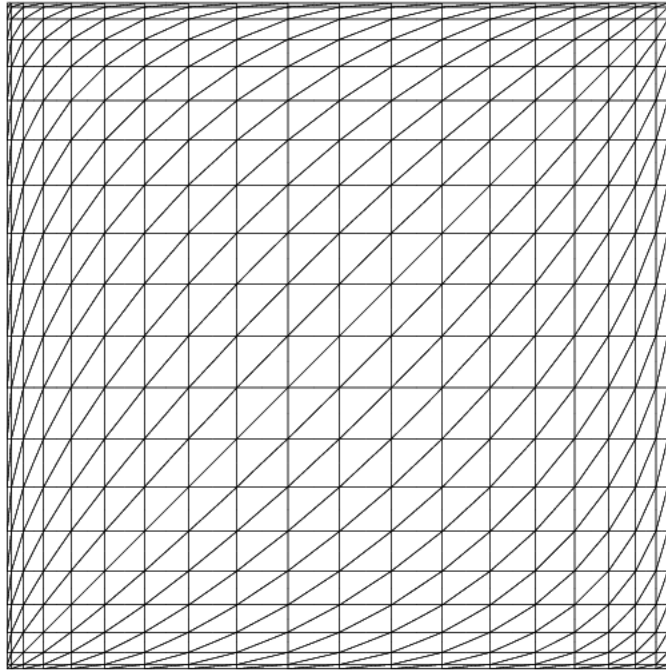


Figure 5: The graded mesh used for the driven cavity.

- For values of R “sufficiently near” R_0 , the approximation error can be decreased by increasing n ;
- For any fixed value of n , the error tends to rise as ΔR increases, and as a power of $n + 1$.

For the following table, the base parameter value $R_0 = 100$ was chosen, at which point the full basis solution $u_{FL}(100)$ was computed, along with the first 5 Taylor vectors and reduced basis vectors. Then, for a sequence of increasing parameter values R , the full basis solution $u_{FL}(R)$ was computed, as well as the Taylor approximant $u_{Tay}(R)$ and the reduced basis solution $u_{RB}(R)$. The relative error of the Taylor approximant was computed as the ratio

$$\frac{\|u_{Tay}(R) - u_{FL}(R)\|}{\|u_{FL}(R)\|} \quad (66)$$

with a similar ratio used for the reduced basis. Both approximation methods were studied with the number of vectors varied from 0 to 5. The results are summarized in Table (1).

Table 1: Comparison of Taylor and Reduced Basis Approximation Errors.

Value of R:	150	200	250	300	350	400
$\ u_{FL}\ $	0.266	0.271	0.274	0.278	0.280	0.282
Taylor						
N_{Tay}						
0	1.10e-1	1.99e-1	2.68e-1	3.21e-1	3.64e-1	3.97e-1
1	3.04e-2	9.98e-2	1.85e-1	2.78e-1	3.75e-1	4.75e-1
2	9.66e-3	6.79e-2	1.96e-1	3.95e-1	6.64e-1	1.00
3	2.56e-3	3.28e-2	1.38e-1	3.71e-1	7.86e-1	1.44
4	1.28e-3	3.41e-2	2.09e-1	7.16e-1	1.82	3.87
5	3.05e-4	1.69e-2	1.69e-1	8.31e-1	2.79	7.34
Reduced Basis						
N_{RB}						
0	1.10e-1	1.99e-1	2.68e-1	3.21e-1	3.64e-1	3.97e-1
1	3.15e-2	1.05e-1	2.01e-1	3.10e-1	4.32e-1	5.60e-1
2	5.64e-3	3.14e-2	7.08e-2	1.13e-1	1.53e-1	1.90e-1
3	1.56e-3	1.51e-2	4.64e-2	9.21e-2	1.48e-1	2.11e-1
4	3.60e-4	5.90e-3	2.21e-2	4.71e-2	7.75e-2	1.08e-1
5	7.86e-5	2.31e-3	1.18e-2	3.14e-2	6.07e-2	9.86e-2

There are several things to note about the data in this table. First, note that for the Taylor data, we see that increasing n reduces the relative error for $R = 150$ and $R = 200$ but has little effect at $R = 250$ and actually begins to increase the error thereafter. This suggests that we can only make accurate Taylor approximations close to the base parameter value.

By contrast, the reduced basis data seems more robust. Even at $R = 400$, we can increase the value of n to decrease the relative error. Moreover, the error made by the reduced basis is generally less than that made by the Taylor polynomial for values of n exceeding 1.

This data tends to confirm the suspicion that the Taylor approximating space is itself good, but that the prescribed coefficients can be improved as we move away from the base parameter value.

Certain points are raised by the graphical comparison of the approximation behavior for the Taylor and reduced basis schemes. First, we note that the Taylor scheme is only guaranteed to have good approximation properties in some “small” neighborhood of unspecified size. Yet the graphs suggest that the neighborhood of $R = 100$ extends to the right at least to $R = 200$. How can we explain such a large range for the approximation? It turns out that an answer is fairly easy to find; the solution curve for this problem is surprisingly “flat”, as evidenced by the l_∞ norms of the velocity sensitivities. The zero order sensitivity is

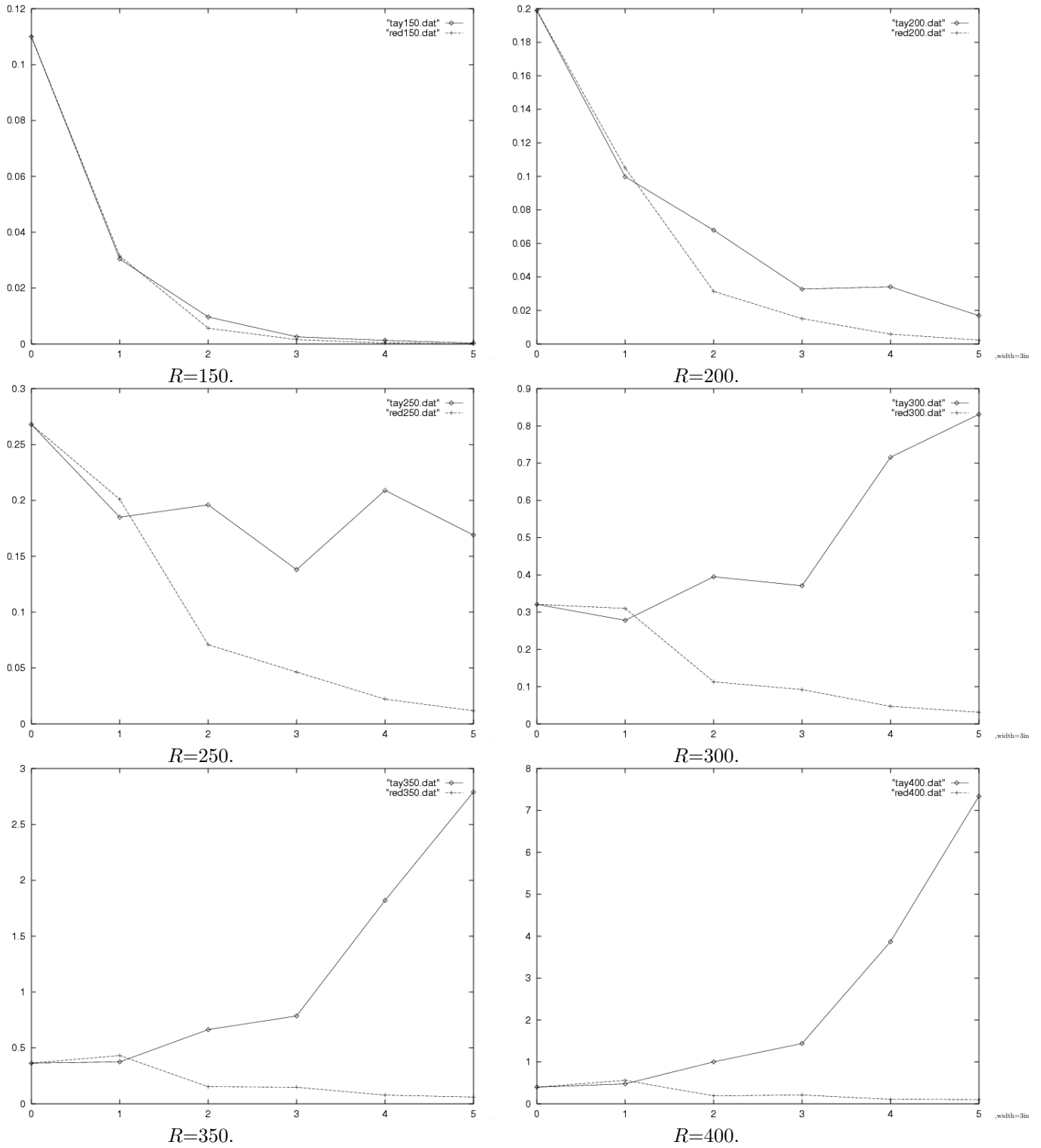


Figure 6: The relative error versus approximation order N .
The value of R varies from 150 to 400.
Taylor method is solid, reduced basis dashed.

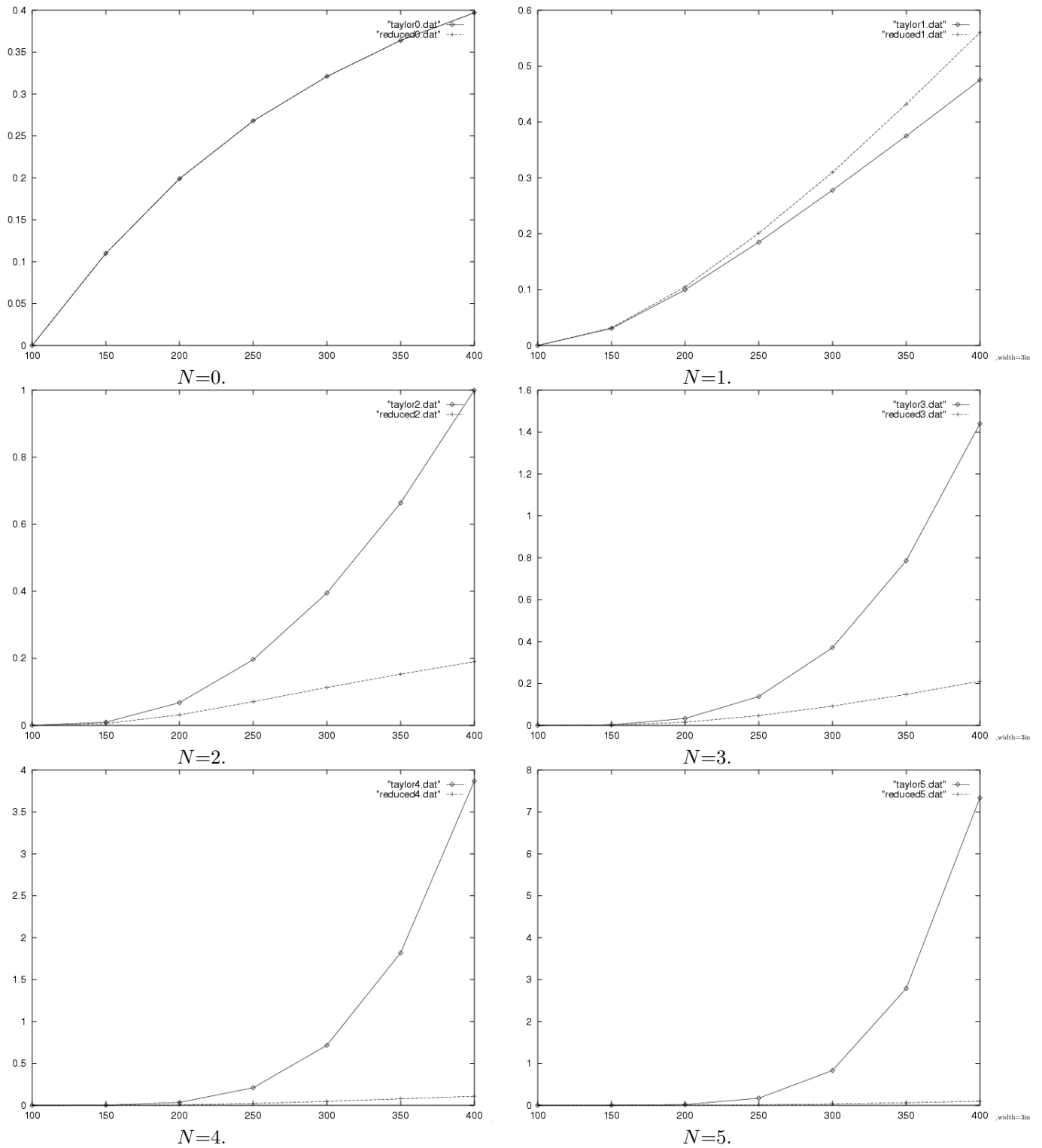


Figure 7: The relative error with increasing R . The approximation order varies from $N=0$ to 5. Taylor method is solid, reduced basis dashed.

500 times larger than the first order, which is 100 times larger than the second order, and the norms continue to decrease with higher order. This suggests that we must take a very large step ΔR indeed before the higher order sensitivities will be multiplied by scale factors large enough to affect the zero order sensitivity.

Table 2: l_∞ Norms of Velocity Sensitivities at $R=100$.

Order:	Norm
0	1.00
1	1.65e-3
2	2.01e-5
3	4.08e-7
4	9.63e-9
5	4.23e-10

Note that this result is a special property of the particular Navier Stokes problem being solved. For problems which are much more nonlinear, we would expect at least a few of the early order derivatives to be of comparable magnitude to the zero order sensitivity, and this would in turn make Taylor approximation more liable to early breakdown.

It is not so easy to answer a similar question about the reduced basis method. If the Taylor approximation breaks down after a while, why does the Taylor space itself still provide a good approximation? We see from the graphs that, at least for the basis solution at $R=100$, the reduced basis method is able to produce approximations still valid at much higher Reynolds numbers than the Taylor method; moreover, the low order approximations can be improved by increasing the approximation order N_{RB} , which is not true for the Taylor method after about $R=200$.

Researchers have been able to show analytically why the reduced basis method produces a good result when quite near the base solution, but have not been able to address the question of why the approximation can be extended over such a wide range.

15 Flow Optimization Using a Reduced Basis

The problem of flow optimization requires the determination of parameter values that specify a flow which in turn produces the “best” value of some special scalar quantity. For our purposes, we may pose this problem as the search for the parameter value λ^* for which the corresponding flow $u(\lambda^*)$ produces the minimal value of the *cost function* $\mathcal{J}(u, \lambda)$.

To efficiently solve this problem using optimization software, it is necessary to evaluate both \mathcal{J} and the partial derivatives \mathcal{J}_u and \mathcal{J}_λ for many argument values (u, λ) .

As a test of these ideas, an optimization problem was set up and solved in the usual way and then by reduced basis methods. The formulation began with the driven cavity problem. The parameter, as before, was the Reynolds number R . The cost functional was the integral of the square of the difference between the computed and desired flow profiles along the vertical bisector of the region.

$$J(\lambda) = \int_{x=0.5} (u(x, y) - u^*(x, y))^2 + (v(x, y) - v^*(x, y))^2 dy \quad (67)$$

The desired flow profile data (u^*, v^*) was generated by setting $R^* = 100$ and solving for the corresponding flow. The optimization procedure was then begun with a starting guess $R_0 = 1$. Table 3 lists the sequence of iterates produced during this optimization.

Now the optimization was repeated, using the reduced basis. A full solution was computed at $R = 1$, the reduced basis was set up there, and then the reduced problem was used to evaluate the cost functional. By this means, the optimization code was able to reach $R = 95$, at which point it signaled that no further

Table 3: Optimization Results for Full System.

Step	R	$J(R)$
0	1.0000	4.99e-3
1	1.0001	4.99e-3
2	1.0005	4.99e-3
3	1.0011	4.99e-3
4	1.0051	4.99e-3
5	1.0111	4.99e-3
6	1.0511	4.98e-3
7	1.1111	4.98e-3
8	1.5120	4.93e-3
9	2.1123	4.88e-3
10	6.1167	4.49e-3
11	12.0432	3.93e-3
12	51.7670	1.13e-3
13	100.7526	2.55e-7
14	100.0962	4.19e-9
15	100.0002	1.68e-14
Convergence		

progress could be made. This was taken to be a signal that the reduced basis should be regenerated. The optimization was restarted from this point, and quickly converged to the desired minimizing value of R .

16 Boundary and Side Conditions in the Reduced System

16.1 Introduction

In the discussion of the reduced basis method, we have concentrated on developing the analogy between the full and reduced systems in terms of a simple difference in the choice of the approximating space for a finite element formulation.

However, we have glossed over a certain asymmetry in this comparison: the side and boundary conditions associated with the full system seem to become “natural” conditions in the reduced system. Thus, in the full system, we have the momentum equations, the continuity equation, and the various boundary conditions, but in the reduced system, only the momentum equations show up explicitly.

We will now discuss the reasons that we were able to ignore. For convenience, we will allow the term “side conditions” to include what we would normally separately designate as “boundary conditions”.

16.2 The Simplest Side Conditions

In the simple state systems discussed so far, the side conditions have the form

$$\mathcal{B}(u) = g \tag{68}$$

where the function \mathcal{B} is actually linear in u , and g depends only on x and y , but not on the parameter R or the state variable u . For this simple problem, the reduced basis solution is the sum of a fixed solution that satisfies the side conditions, plus a linear combination of solutions that satisfy the linearized conditions.

The linearity of \mathcal{B} then tells us that the reduced basis solution will satisfy the conditions (68).

We have generally treated the continuity equation of the Navier Stokes system as a state equation; it makes more sense to treat it as a side condition, because we can now explain why it “disappears” as well,

Table 4: Optimization Results for Reduced System.

Step	R	$J(R)$
0	1.0000	4.99e-3
5	1.0350	4.98e-3
6	1.1076	4.98e-3
7	1.5083	4.94e-3
8	2.1083	4.88e-3
9	6.1113	4.49e-3
10	12.0375	3.93e-3
11	51.7543	1.11e-3
12	102.7267	5.19e-5
13	94.8888	1.99e-5
14	95.30002	1.99e-5
15	95.30091	1.99e-5
20	95.30003	1.99e-5
	Restart	
21	95.30003	1.00e-5
25	95.30050	1.00e-5
30	95.51940	9.14e-6
31	95.77294	8.13e-6
32	97.47383	2.89e-6
33	99.63274	6.10e-8
34	99.99644	5.79e-12
	Convergence	

in the reduced system. It's simply that the sum of flows satisfying the continuity equation will itself satisfy the continuity equation.

Note that, assuming that the basic solution and sensitivities exactly satisfy the state system, the form of the reduced basis approximation guarantees that *every* element of the reduced basis space satisfies all the side conditions; thus, the prediction for the reduced solution at a new value of the parameter must satisfy the side conditions, simply because it is an element of the reduced space.

And since the reduced Newton iteration stays within the reduced space, every Newton iterate, and the final accepted point, satisfy the side conditions, automatically. Thus we see that the reduced basis method guarantees the satisfaction of simple side conditions automatically and implicitly.

This argument applies to the simple case where the side conditions are linear in the state variable and independent of the parameter. In the next section, we discuss the more general case where the side conditions can vary with the parameter.

16.3 Prediction for Conditions Dependent on the Parameter

Now let us suppose that the side conditions depend in some way on the parameter R . Then we may write our set of side conditions as

$$\mathcal{B}(R, u) = g(R). \tag{69}$$

Now the form of the first order sensitivity equations will be:

$$\mathcal{B}_u(R, u) u_R = -\mathcal{B}_R(R, u) + g_R(R). \tag{70}$$

Now let us suppose that we have a pair (R_0, u_0) which satisfy the boundary conditions (69), and that we only wish to use the first sensitivity to predict the solution at $R + \Delta R$.

Then our predicted solution would be:

$$u(R_0 + \Delta R) \approx u_0 + \Delta R u_R. \quad (71)$$

Now the residual in the side conditions at $R + \Delta R$ can be written as:

$$\begin{aligned} \mathcal{B}(R_0 + \Delta R, u(R_0 + \Delta R)) - g(R) = & \\ & \mathcal{B}(R_0, u(R_0)) + \mathcal{B}_u(R_0, u(R_0)) u_R(R_0) \Delta R + \mathcal{B}_R(R_0, u(R_0)) \Delta R \\ & - g(R_0) - g_R(R_0) \Delta R + O((\Delta R)^2) \end{aligned} \quad (72)$$

but we may regroup the quantities on the right hand side to become:

$$\begin{aligned} = & \mathcal{B}(R_0, u(R_0)) - g(R_0) \\ & + \Delta R (\mathcal{B}_u u_R + \mathcal{B}_R(R_0, u(R_0)) - g_R(R_0)) \\ & + O((\Delta R)^2). \end{aligned} \quad (73)$$

Now the first group disappears by the definition of u_0 , and the second group by the definition of u_R . If in fact the dependence of \mathcal{B} and f on R is strictly linear, then the error term $O((\Delta R)^2)$ drops out, and the predicted point exactly satisfies the side conditions; otherwise, it satisfies them to order $O((\Delta R)^2)$. If we increase the order of our reduced basis, then the same argument can be extended: the error in the predicted point is either $O((\Delta R)^{(N_{RB}+1})$, or it vanishes altogether.

The case where the parameter R enters the side conditions linearly is common, and we will assume at this point that this is in fact the case for the problems we will look at. Then, assuming we have used at least the first order reduced basis vector, our predicted point exactly satisfies the side conditions. Our next question is, naturally enough, what happens to the satisfaction of the side conditions when the Newton method is applied to the predicted point to force satisfaction of the reduced state equations?

16.4 Correction for Conditions Dependent on the Parameter

Now let us consider the effect of the application of the Newton iteration to the satisf

Now we need to determine whether the Newton iteration preserves or destroys the satisfaction of the side conditions. We really only need to consider one step of the iteration. So suppose that u_0 is an estimated value for the reduced basis solution at the parameter R , which satisfies the side conditions (69).

16.5 Algorithmic Implications

It should be clear that if we are only interesting in solving problems with the simplest side conditions, as expressed in Equation (68), then our strategy can be quite simple. We generate the reduced basis, whose first column is u^0 , which satisfies the nonhomogenous side conditions. We then remove this vector from the reduced basis, forming the sensitivity matrix S from the vectors $u^1, u^2, \dots, u^{N_{RB}}$, orthogonalizing S to form the reduced basis matrix RB , so that a typical element of the reduced space may be represented as:

$$u = u^0 + \sum_{i=1}^{N_{RB}} c_i u^i \quad (74)$$

We know that u^0 satisfies the boundary conditions, and we are free to determine the unknown coefficients c_i via the reduced finite element equations.

However, suppose now that our side conditions have the form of Equation (69). Let us suppose further that for some set of coefficients d_i , the linear combination

$$u_{BC} = \sum_{i=0}^{N_{RB}} d_i u^i \quad (75)$$

will satisfy the side conditions. Then in order to proceed, we must “factor out” the vector u_{BC} from the sensitivity matrix S , leaving only basis vectors that are orthogonal to that direction, so that the side condition is preserved. This case subsumes the previous one, if we take u_{BC} as equal to u^0 , and $d_0 = 1$ and all other $d_i = 0$. To remove this vector from the basis, we simply perform a QR factorization of the matrix u_{BC} , u^0 , ..., $u^{N_{RB}}$ and then discard the column with the smallest element on the diagonal of the R factor.

References

- [1] E Anderson *et al*, **LAPACK User’s Guide**, Society for Industrial and Applied Mathematics, Philadelphia, 1992.
- [2] A Chan and K Hsiao, *Nonlinear Analysis Using a Reduced Number of Variables*, Computer Methods in Applied Mechanics and Engineering, volume 52, 1985, pages 899-913.
- [3] Jim Fink and Werner Rheinboldt, *On the Error Behavior of the Reduced Basis Technique for Nonlinear Finite Element Approximations*, Zeitschrift für Angewandte Mathematik und Mechanik, volume 63, 1983, pages 21-28.
- [4] Max Gunzburger, **Finite Element Methods for Viscous Incompressible Flows**, Academic Press, San Diego, 1989.
- [5] Max Gunzburger and Janet Peterson, *Predictor and Steplength Selection in Continuation Methods for the Navier Stokes Equations*, Computers and Mathematics with Applications, volume 22, number 8, 1991, pages 73-81.
- [6] Dennis Nagy, *Modal Representation of Geometrically Nonlinear Behavior by the Finite Element Method*, Computers and Structures, volume 10, 1979, pages 683-688.
- [7] Ahmed Noor, *Recent Advances in Reduction Methods for Nonlinear Problems*, Computers and Structures, volume 13, 1981, pages 31-44.
- [8] Janet Peterson, *The Reduced Basis Method for Incompressible Viscous Flows*, SIAM Journal of Scientific and Statistical Computing, volume 10, 1989, pages 777-786.
- [9] T A Porsching, *Estimation of the Error in the Reduced Basis Method Solutions of Nonlinear Equations*, Mathematics of Computation, Volume 45, Number 172, October 1985, pages 487-496.
- [10] T A Porsching and M Lin Lee, *The Reduced Basis Method for Initial Value Problems*, SIAM Journal of Numerical Analysis, Volume 24, Number 6, December 1987, pages 1277-1287.