Model Order Reduction Techniques
Problem Set 2 — RB for Linear Affine Elliptic Problems

Christophe Prud'homme

1 Problem Statement — Design of a Thermal Fin

We consider the problem of designing a thermal fin described in Problem Set 1. In PS1 we looked at some thoeretical issues (weak formulation and optimization formulation, convergence of the reduced basis approximation) and derived the necessary reduced basis quantities, i.e., expressions for $A_N(\mu)$, F_N , and L_N . This problem set is devoted to implementing the reduced basis approximation and solving a simple design problem.

1.1 Part 1 – Reduced Basis Approximation

The point of departure for the reduced basis approximation is a high – dimensional finite element truth discretization. In the offline stage we require the finite element solution to build the reduced basis and we thus also need the FE matrices. In this problem set we skip the FE assembly step and provide all of the necessary data for use in MATLAB (see Appendix 1).

We saw in class that the reduced basis solution $u_N(\mu) \in \mathbb{R}^N$ satisfies the set of $N \times N$ linear equations,

$$A_N(\mu)u_N(\mu) = F_N; \tag{1}$$

and that the outputis given by

$$T_{root\,N}(\mu) = L_N^T u_N(\mu). \tag{2}$$

We derived expressions for $A_N(\mu) \in \mathbb{R}^{N\times N}$ in terms of $A_N(\mu)$ and Z, $F_N \in \mathbb{R}^N$ in terms of F_N and Z, and $L_N \in \mathbb{R}^N$ in terms of L_N and Z; here Z is an $\mathcal{N} \times N$ matrix, the jth column of which is $u_N(\mu j)$ (the nodal values of $u_N(\mu j)$). Finally, it follows from affine parameter dependence that $A_N(\mu)$ can be expressed as

$$A_N(\mu) = \sum_{q=1}^{Q} \Theta^q(\mu) A_N^q. \tag{3}$$

The goal is to implement an offline/ online version of the reduced – basis method following the computational decomposition indicated below.

- Offline
  1. Choose N.
  2. Choose the sample S_N .
  3. Construct Z.
  4. Construct $A_N^q, q = 1, \ldots, Q; F_N;$ and L_N.
- Online
  1. Form $A_N(\mu)$ from (3).
  2. Solve $A_N(\mu)u_N(\mu) = F_N$.
  3. Evaluate the output $T_{root\,N}(\mu)$ from (2).

1 The idea is that the offline stage is done only once, generating a small datafile with the $A_N^q, q = 1, \ldots, Q$, F_N, and L_N; the on-line stage then accesses this datafile to provide real-time response to new μ queries. For the required off-line finite element calculations in this and the following questions, you should first use the coarse triangulation $\mathcal{T}_{h,\text{coarse}}$.
a) Show that the operation count for the on-line stage of your code is independent of $\mathcal{N}$. In particular show that the operation count (number of floating-point operations) for the on-line stage, for each new μ of interest, can be expressed as

$$c_1 N^{\gamma_1} + c_2 N^{\gamma_2} + c_3 N^{\gamma_3}, \tag{4}$$

for $c_1, c_2, c_3, \gamma_1, \gamma_2$, and γ_3 independent of N. Give values for the constants $c_1, c_2, c_3, \gamma_1, \gamma_2$, and γ_3.
b) We first consider a one parameter ($P = 1$) problem. To this end, we keep the Biot number fixed at $Bi = 0.1$ and assume that the conductivities of all fins are equivalent, i.e., $k_1 = k_2 = k_3 = k_4$, but are allowed to vary between 0.1 and 10 we thus have $\mu \in D = [0.1, 10]$. The sample set S_N for $N_{max} = 8$ is given in the datafile `RB_sample.sample1`.

1. Generate the reduced basis matrix Z and all necessary reduced basis quantities. You have two options: you can use the solution "snapshots" directly in Z or perform a Gram-Schmidt orthonormalization to construct Z (Note that you require the X – inner product to perform Gram-Schmidt; here, we use $(\cdot, \cdot)_X = a(\cdot, \cdot; \mu)$, where $\mu = 1$ all conductivities are 1 and the Biot number is 0.1). Calculate the condition number of $A_N(\mu)$ for $N = 8$ and for $\mu = 1$ and $\mu = 10$ with and without Gram – Schmidt orthonormalization. What do you observe? Solve the reduced basis approximation (where you use the snapshots directly in Z) for $\mu_1 = 0.1$ and $N = 8$. What is $u_N(\mu_1)$? How do you expect $u_N(\mu_2)$ to look like for $\mu_2 = 10.0$? What about $\mu_3 = 1.0975$? Solve the Gram – Schmidt orthonormalized reduced basis approximation for $\mu_1 = 0.1$ and $\mu 2 = 10$ for $N = 8$. What do you observe? Can you justify the result? For the remaining questions you should use the Gram – Schmidt orthonormalized reduced basis approximation.

2. Verify that, for $\mu = 1.5$ (recall that Biot is still fixed at 0.1) and $N = 8$, the value of the output is $T_{root\,N}(\mu) = 1.53107$.

3. We next introduce a regular test sample, $\Xi_{test} \subset D$, of size $ntest = 100$ (in MATLAB you can simply use `linspace(0.1, 10, 100)` to generate Ξ_{test}). Plot the convergence of the maximum relative error in the energy norm $\max_{\mu \in \Xi_{test}} |||u(\mu) - u_N(\mu)|||_\mu / |||u(\mu)|||_\mu$ and the maximum relative output error max $\mu \in \Xi_{test} |T_{root}(\mu) - T_{root} N(\mu)| / T_{root}(\mu)$ as a function of N (use the MATLAB command `semilogy` for plotting).

4. Compare the average CPU time over the test sample required to solve the reduced basis online stage with direct solution of the FE approximation as a function of N.

5. What value of N do you require to achieve a relative accuracy in the output of 1%. What savings in terms of CPU time does this % correspond to?

6. Solve problems b) 3. to 5. using the medium and fine FE triangulation. Is the dependence on $\mathcal{N}$ as you would anticipate?

c) We now consider another one parameter ($P = 1$) problem. This time, we assume that the conductivities are fixed at $\{k_1, k_2, k_3, k_4\} = \{0.4, 0.6, 0.8, 1.2\}$, and that only the Biot number, Bi, is allowed to vary from 0.01 to 1. The sample set S_N for $N_{max} = 11$ is given in the datafile `RB_sample.sample2`. Generate an orthonormal Z from the sample set using the medium triangulation.

- Verify that, for $\mu_0 = 0.4, 0.6, 0.8, 1.2, 0.15$, i.e. $Bi = 0.15$, the value of the output is $T_{root\,N}(\mu 0) = 1.51561$.

- We next introduce a regular test sample, $\Xi_{test} \subset D$, of size $ntest = 100$ (in MATLAB you can simply use `linspace(0.01, 1, 100)` to generate Ξ_{test}). Plot the convergence of the maximum relative error in the energy norm $\max_{\mu \in \Xi_{test}} |||u(\mu) - u_N(\mu)|||_\mu / |||u(\mu)|||_\mu$ and the maximum relative output error $\max_{\mu \in \Xi_{test}} |T_{root}(\mu) - T_{root\,N}(\mu)| / T_{root}(\mu)$ as a function of N (use the MATLAB command `semilogy` for plotting).

- The Biot number is directly related to the cooling method; higher cooling rates (higher Bi) imply lower (better) T_{root} but also higher (worse) initial and operational costs. We can thus define (say) a total cost function as

$$C(Bi) = Bi + T_{root}(Bi), \tag{5}$$

  minimization of which yields an optimal solution. Apply your (online) reduced – basis approx – imation for $T_{root\,N}$ (that is, replace $T_{root}(Bi)$ in (5) with $T_{root\,N}(Bi)$) to find the optimal Bi. Any (simple) optimization procedure suffices for the minimization.

d) Finally, we consider a two parameter ($P = 2$) problem where the conductivities are assumed to be equivalent, i.e., $k_1 = k_2 = k_3 = k_4$, but are allowed to vary between 0.1 and 10; and the Biot number, Bi, is allowed to vary from 0.01 to 1. The sample set S_N for $N_{max} = 46$ is given in the datafile `RB_sample.sample3`. Generate an orthonormal Z from the sample set using the coarse triangulation.

1. 1. We next introduce a regular grid, $\Xi_{test} \subset D$, of size $ntest = 400$ (a regular 20×20 grid). Plot the convergence of the maximum relative error in the energy norm $\max_{\mu\in\Xi_{test}} |||u(\mu) - u_N(\mu)|||_\mu / |||u(\mu)|||_\mu$ and the maximum relative output error $max_{\mu\Xi_{test}} |T_{root}(\mu) - T_{root\,N}(\mu)|/T_{root}(\mu)$ as a function of N (use the MATLAB command semilogy for plotting).

1.2 Appendix 1 – Finite Element Method Implementation

For the implementation of the reduced basis method, the finite element matrices for three possible triangulations of the fin problem are provided. To obtain the required matlab data, download the file `PS2_matlab.zip` from the course web site und unzip it. There are three .mat files: `FE_matrix.mat` contains the FE matrices, `FE_grid.mat` contains the triangulation data, and `RB_sample.mat` contains the samples you should use initially (later on you will generate samples yourselves using a greedy procedure). To load the FE matrices in the MATLAB workspace:

```
>> load FE_matrix
```

This creates one variable named `FE_matrix` with three fields coarse, medium, and fine. Each of these fields contains a cell array `Ahq` of size 6×1 and the load vector `Fh`. Each cell of `Ahq` contains the parameter – independent FE matrix $A_N^q, q = 1, ..., 6$; here $q = 1, ..., 4$ corresponds to the "submatrices" of fins $1, ..., 4$, with conductivities $k_i, i = 1, ..., 4$, respectively; $q = 5$ corresponds to the "submatrix" of the central post with conductivity $k_0 = 1$; and $q = 6$ corresponds to the "submatrix" of the line integral over the "surface" of the fin (without Γ_{root}). To load the reduced basis samples SN in the MATLAB workspace:

```
>> load RB_sample
```

This creates one variable named `RB_sample` with fields sample1, sample2, and sample3, corresponding to the two $P = 1$ and the $P = 2$ cases described in the problem statement. Note that you require the triangulation only for plotting the FE solution (see below). The following detailed information about the triangulation is just included to give you an impression concerning the data required if you would like to set up the FE matrices from scratch. To load the triangulation data in the MATLAB workspace:

```
>> load FE_grid
```

This creates one variable `FE_grid` with three fields coarse, medium, and fine. Each of these fields is a different triangulation $\mathcal{T}_h$ for the fin problem. More specifically

- coarse defines $\mathcal{T}_{h,coarse}$, with 1333 nodes, and 2095 elements.
- medium defines $\mathcal{T}_{h,medium}$, with 4760 nodes, and 8380 elements, and
- fine defines $\mathcal{T}_{h,fine}$, with 17889 nodes, and 33520 elements.

Each of these variables is of type struct, with four different fields.

```
>> coarse
coarse =
     nodes:  1333
     coor:  [1333x2 double]
     elements:  2095
     theta:  1x7 cell
```

Description of the fields: (assume that we are using the coarse triangulation)

- `nodes`: The number of nodes in the triangulation.
- `coor`: Two-dimensional matrix with size (nodes×2), where each row i has the x and y coordinates for node i. For example, the location of node 49 can be determined by two coordinates. The coordinate in the x – direction would be `coarse.coor(49,1)` and in the y – direction `coarse.coor(49,2)`.
- `elements`: The number of elements in the triangulation.
- `theta`: The adjacency matrix $\Theta(k,\alpha)$ which defines the local – to – global mapping required in the elemental assembly procedure. Since we have regions with different physical properties, for each region a seperate adjacency matrix is provided. The regions considered are
  - Region 1: Domain $\Omega_1, \Theta_1(k,\alpha)$ =`coarse.theta1,`
  - Region 2: Domain $\Omega_2, \Theta_2(k,\alpha)$ =`coarse.theta2,`
  - Region 3: Domain $\Omega_3, \Theta_3(k,\alpha)$ =`coarse.theta3,`
  - Region 4: Domain $\Omega_4, \Theta_4(k,\alpha)$ =`coarse.theta4,` Region 5: Domain $\Omega_0, \Theta_5(k,\alpha)$ =`coarse.theta5.`

  For each of these regions i, the index k varies in the range $k \in 1, ..., n_i$, where n_i are the number of elements in region i. For example element 12 in region 3 is has the global nodes ν_1 =`coarse.theta3(12,1)`, ν_2 =`coarse.theta3(12,2)`, and ν_3 =`coarse.theta3(12,3)`. In addition, for the treatment of the boundary conditions, the boundary is divided into two sections. The first is $\Gamma \backslash \Gamma_{root}$, where Robin boundary conditions are applied; the second is Γ_{root}, where the incoming heat flux is applied. For each segment in these sections, the associated global nodes are provided.
  - Section 1: $\Gamma \backslash \Gamma_{root}, \kappa_1(m,\alpha)$ =`coarse.theta6`,
  - Section 2: $\Gamma_{root}, \kappa_2(m,\alpha)$ =`coarse.theta7.`

  For each of the sections i, the index m varies in the range $m \in 1, ..., s_i$, where the s_i are the number of segments in section i. As an example, to find the nodes ν_1, and ν_2 for segment 5 in the first section, we would use ν_1 =`coarse.theta6(5,1)`, and ν_2 =`coarse.theta6(5,2).`

To plot the temperature distribution, `plotsolution.m` can be used. If $z \equiv u_h$ is the vector with the computed temperature values for each of the nodes, then a contour plot of the temperature distribution can be obtained by

```
>> plotsolution(FE_grid.coarse, z)
```

The first argument is the mesh used in the calculation of z, and the second is the solution vector z.

For the storage of the finite element matrices, use MATLABs sparse matrix data structure. Also, for the solution of the resulting linear systems, use the default solution methods provided in MATLAB, i.e.. use

```
>> u=A\F
```

to solve for the FEM solution u.

Acknowledgment: I would like to thank Prof. A.T. Patera and Debbie Blanchard for providing some of the material for this problem set and for allowing us to use it.