University of Helsinki
Department of Mathematics and Statistics

Computational statistics 2 — assignment

This assignment contains 1 problem. You *may* use your books and notes. You are required to show your work on each problem of this assignment. The following rules apply:

- **No corrections will be made once the assignment is distributed.** If there is a typo or ambiguity in a question, state your assumption and answer accordingly.
- **Organize your report** ("fullname_report_COMPSTAT2014_2.pdf") in a reasonably neat and coherent way. Algebraic work and source code scattered all over without a clear ordering will receive very little credit. Provide a title page with your report that contains your full name, email address and student number. Put the same information on top of every page of your report, in case the pages become separated.
- **All R code files need to be handed in** and named "fullname_code1_COMPSTAT2014_2.R", "fullname_code2_COMPSTAT2014_2.R", etc. Put your full name, email address and student number in the source code header comments.
- **Mysterious or unsupported answers will not receive full credit**. A correct answer, unsupported by calculations, explanation, source code or algebraic work will receive no credit. An incorrect answer supported by substantially correct calculations and explanations might still receive partial credit.
- If you are graduating or returning to your home university at the end of the semester, please write "GRADUATING" or "EXCHANGE" on the title page of your report.
- **The deadline for returning your results is Saturday, February 28, 2015**. ZIP compress all files and folders and email the archive named "fullname_results_COMPSTAT2014_2.zip" to christian.benner@helsinki.fi.

Task: Implement the algorithm by Bai et al. to compute $\mathrm{tr}\left(\boldsymbol{A}^{-1}\right)$, that is, the trace of the inverse of a matrix. The algorithm to compute $\mathrm{tr}\left(\boldsymbol{A}^{-1}\right)$ can be described by the following pseudocode.

Algorithm 1: Monte Carlo estimation of the trace of the inverse of a matrix (Bai et al.)

input : Positive definite matrix $\boldsymbol{A}$ and total number of Monte Carlo samples n
output: An estimate of the trace of $\boldsymbol{A}^{-1}$

Compute the extreme eigenvalues a and b of $\boldsymbol{A}$. (Hint: This can be done by for instance directly exploiting the properties of $\boldsymbol{A}$ or using the Power method)

for $i = 1$ to n **do**
  Apply `Lanczos()` to matrix $\boldsymbol{A}$. Obtain an estimate of $\mathrm{tr}(\boldsymbol{A}^{-1})$ using `Gauss-Radau()`.
end
Average all n estimate of $\mathrm{tr}\left(\boldsymbol{A}^{-1}\right)$.

Procedure `Lanczos()`

1. Start the Lanczos algorithm with a vector $\boldsymbol{x}_0$ that contains independent samples from a random variable which takes the values 1 and -1 each with probability 0.5. Normalize the vector $\boldsymbol{x}_0$.
2. In each iteration j of the Lanczos algorithm, obtain the eigenvalues $\boldsymbol{\lambda}$ and eigenvectors $\boldsymbol{V}$ of the symmetric tridiagonal matrix $\boldsymbol{T}_j$ using the lower and upper bound a and b.
3. Compute $I = \sum_{k=1}^{j} \boldsymbol{V}_{1k}^2 / \lambda_k$ using the normalized eigenvectors.
4. Stop the Lanczos algorithm if the absolute change in I is less than some convergence criterion.
5. **return** the diagonal $\boldsymbol{\alpha}$ and super-diagonal $\boldsymbol{\beta}$ of $\boldsymbol{T}_j$ as well as the dimension of $\boldsymbol{T_j}$.

Procedure `Gauss-Radau()`

// Upper bound on trace of $\boldsymbol{A}^{-1}$
1. Solve the tridiagonal system $(\boldsymbol{T}_j - a\boldsymbol{I})\boldsymbol{\delta} = \beta_j^2 \boldsymbol{e}_j$.
2. Obtain the eigenvalues $\boldsymbol{\lambda}$ and eigenvectors $\boldsymbol{V}$ of the symmetric tridiagonal matrix $\tilde{\boldsymbol{T}}_j = \begin{bmatrix} \boldsymbol{T}_j & \beta_j \boldsymbol{e}_j \\ \beta_j \boldsymbol{e}_j^{\mathrm{T}} & a + \delta_j \end{bmatrix}$ using the lower and upper bound a and b.
3. Compute $U = \sum_{k=1}^{j} \boldsymbol{V}_{1k}^2 / \lambda_k$ using the normalized eigenvectors.

// Lower bound on trace of $\boldsymbol{A}^{-1}$
4. Solve the tridiagonal system $(\boldsymbol{T}_j - b\boldsymbol{I})\boldsymbol{\delta} = \beta_j^2 \boldsymbol{e}_j$.
5. Obtain the eigenvalues $\boldsymbol{\lambda}$ and eigenvectors $\boldsymbol{V}$ of the symmetric tridiagonal matrix $\tilde{\boldsymbol{T}}_j = \begin{bmatrix} \boldsymbol{T}_j & \beta_j \boldsymbol{e}_j \\ \beta_j \boldsymbol{e}_j^{\mathrm{T}} & b + \delta_j \end{bmatrix}$ using the lower and upper bound a and b.
6. Compute $L = \sum_{k=1}^{j} \boldsymbol{V}_{1k}^2 / \lambda_k$ using the normalized eigenvectors.
7. **return** $0.5 \times (L + U) \times \dim(A)$.

Problem 1: Generate 3 matrices using $n = 1000, 5000, 10000$.

```
A <- matrix( runif( n ^ 2, 0, 0.1 ), n, n ) ; A <- t( A ) + A
diag( A ) <- rowSums( A )
fh <- file( sprintf( 'matrix_n%d.bin', n ), 'wb' )
writeBin( as.integer( n ), fh, size = 4 ) ; writeBin( c( A ), fh, size = 4 )
close( fh )
```

Run the above algorithm on the 3 matrices. Report the Monte Carlo point estimate of the trace of the inverse of the matrices as well as confidence intervals as a function of the number Monte Carlo samples. Compare the computing time of the algorithm with another method (e.g. Eigen decomposition).