Computational Assignment II
The Implicitly Restarted Arnoldi Method

Jan Brandts, http://staff.science.uva.nl/∼brandts

In this assignment you will write a MatLab function that implements a simplified version of the Implicitly Restarted Arnoldi Method that was discussed in the Lecture Notes 7-8. We will distinguish between some theoretical questions and some implementation tasks

Part 01. Recall that an *Arnoldi factorization* is an orthonormal basis of $K^{k+1}(A, v)$ represented as the columns of a matrix V_{k+1} together with the upper Hessenberg representation $H_{k+1,k}$ of AV_k on this basis. They are related as follows,

$$AV_k = V_{k+1}H_{k+1,k}. \tag{1}$$

To compute such a factorization, A, v, and k should be given. Then firstly, v needs to be scaled to norm one. The resulting vector will be the first column v_1 of V_{k+1}. Then apply A to find $w = Av_1$. Orthogonalize w to v_1 and normalize the result. This gives the second column v_2 of V_{k+1}. By construction, Av_1 is a linear combination of v_1 and v_2, and the coefficients that determine which linear combination it is, are the coefficients h_{11} and h_{21} of the matrix $H_{k+1,k} = (h_{ij})$. The next step is to compute Av_2, and to orthogonalize it to v_1 and v_2, to normalize the result, and so on.

Part 02. Write a MatLab function

```
[V,H] = ArnoldiInit(A,v)
```

that computes the Arnoldi factorization for given `A` and `v` with $k = 1$. Thus, `V` has two columns and `H` is two by one only (see (9) in Lecture Notes 7-8).

Part 03. Next, write a function that extends a given factorization:

```
[V,H] = ArnoldiPlus(A,V,H,m)
```

In this function, you may assume that the inputs `V` and `H` represent a valid Arnoldi factorization for some $k \geq 1$. This assures that `V` has at least two columns, and `H` at least one. The function should extend the factorization by computing `m` more basisvectors to be added to `V`. The returned matrix `V` should therefore have $n \times (k + m)$ mutually orthonormal columns. The matrix `H` must be $(k + m) \times k + m - 1$.

Part 04. Test your routine on an example. Compute the input factorization of ArnoldiPlus using ArnoldiInit. In your test, use a small matrix, and check if

- `V` and `H` are indeed orthogonal and upper Hessenberg, and
- relation (1) should hold.

Part 05. Test the routine on a symmetric matrix A having the distinct and single eigenvalues $1, 2, \ldots, 20$ and compute the eigenvalue approximations for A in each expansion by one more dimension. This way you should be able to reproduce a triangle of eigenvalue approximations like in (5-7) in Lecture Notes 7-8.

Part 06. Given any value μ in the complex plane, and an Arnoldi factorization of length $k \geq 2$, we have seen that it is possible to compute the Arnoldi factorization of length $k-1$ for the start vector $\hat{v}_1 = \tilde{v}_1/\|\tilde{v}_1\|$ with $\tilde{v}_1 = (A - \mu I)v$ without using any additional matrix vector multiplication with the matrix A again. Write a function for this:

```
[V,H] = ArnoldiMinus(V,H,mu)
```

Again, test this routine on a simple example.

Part 07. Finally, write a program that combines the routines above and that implements the Implicitly Restarted Arnoldi Method as follows:

- At each iteration, list the eigenvalue approximations
- Prompt for the choice between expanding further, or removing an eigenvalue
- Implement this idea graphically: show the approximate eigenvalues in the complex plane. A mouseclick on an eigenvalue should remove it. A mouseclick on a *expand* button should expand the space further.