Bergische Universität Wuppertal

Fachbereich Mathematik und Naturwissenschaften

Institute of Mathematical Modelling, Analysis and Computational Mathematics (IMACM)

Preprint BUW-IMACM 14/04

Andreas Frommer, Stefan Güttel, Marcel Schweitzer

FUNM_QUAD: An implementation of a stable, quadrature based restarted Arnoldi method for matrix functions

February 18, 2014

http://www.math.uni-wuppertal.de

FUNM_QUAD: AN IMPLEMENTATION OF A STABLE, QUADRATURE BASED RESTARTED ARNOLDI METHOD FOR MATRIX FUNCTIONS

ANDREAS FROMMER*, STEFAN GÜTTEL†, AND MARCEL SCHWEITZER*

This note gives a short overview of the `FUNM_QUAD` Matlab code which implements the restarted Arnoldi algorithm described and analyzed in [1] and can be downloaded at `http://www.guettel.com/funm_quad` or `http://www-ai.math.uni-wuppertal.de/SciComp/software/funm_quad.html`.

The code can be used to approximate $f(A)\boldsymbol{b}$, the action of a matrix function on a vector, by a restarted Arnoldi method for an arbitrary (Hermitian or non-Hermitian) matrix A and vector $\boldsymbol{b}$ and a function f with an integral representation of the form

$$f(z) = \int_\Gamma \frac{g(t)}{t-z}\,\mathrm{d}t. \tag{1}$$

For details concerning the algorithm and functions with such integral representations, we refer the reader to [1].

The basic calling sequence of `FUNM_QUAD` is

```
[f,out] = funm_quad(A,b,param),
```

where `A` is a (sparse) quadratic matrix, `b` is a vector of corresponding length and `param` controls various parameters (including the function f) of the algorithm. The output parameter `f` corresponds to the final approximation to $f(A)\boldsymbol{b}$ while `out` contains various other output parameters. In the following, we describe the possible input and output parameters in detail.

Inputs:

- `param.function`: The function f to be evaluated. Predefined functions are `'invSqrt'` for $f(z) = z^{-\frac{1}{2}}$, `'exp'` for $f(z) = e^z$ and `'log'` for $f(z) = \log(1+z)/z$. Other functions can be evaluated by specifying a function handle for the integrand in (1).
- `param.restart_length`: The number of Arnoldi steps performed in each restart cycle.
- `param.max_restarts`: The maximum number of restart cycles to be performed.
- `param.tol`: The error tolerance for numerical quadrature.
- `param.hermitian`: Specifies whether `A` is Hermitian.
- `param.V_full`: Specifies whether the full Arnoldi basis should be stored and returned by the function.
- `param.H_full`: Specifies whether all Hessenberg matrices should be stored and returned by the function.
- `param.exact`: If the exact solution $f(A)\boldsymbol{b}$ is known it can be passed to the function for computation of the error after each cycle.
- `param.stopping_accuracy`: Accuracy at which the algorithm is terminated.
- `param.inner_product`: The inner product used for orthogonalization.

*Department of Mathematics, Bergische Universität Wuppertal, 42097 Wuppertal, Germany, {frommer,schweitzer}@math.uni-wuppertal.de

†School of Mathematics, The University of Manchester, M13 9PL Manchester, United Kingdom, stefan.guettel@manchester.ac.uk

- `param.thick`: Thick-restart function for implicitly deflated restarts. Typically, this will be the function `thick_quad` provided with our code.
- `param.number_thick`: Number of target eigenvalues to be deflated when thick restarts are used.
- `param.min_decay`: Desired rate of linear error reduction. If this rate is no longer achieved, the algorithm terminates.
- `param.reorth_number`: Number of reorthogonalizations in Anroldi's method.
- `param.transformation_parameter`: Parameter used in the integral transformation when dealing with $f(z) = z^{-\frac{1}{2}}$. For details on the choice of this parameter, see [1].
- `param.waitbar`: Specifies whether a waitbar illustrating the progress of the algorithm is shown.

Outputs:

- `out.stop_condition`: Specifies why the algorithm terminated (maximum number of iterations reached, achieved desired accuracy etc.).
- `out.V_full`: Full Arnoldi basis (if desired).
- `out.H_full`: Hessenberg matrices from all restart cycles (if desired).
- `out.time`: CPU time needed for each restart cycle.
- `out.thick_interpol`: Interpolation nodes (Ritz values) from each restart cycle.
- `out.thick_replaced`: Additional interpolation nodes from thick restart procedure for each cycle (if used).
- `out.num_quadpoints`: Number of quadrature points used for evaluating the error function in each restart cycle.
- `out.appr`: Arnoldi approximation after each restart cycle.
- `out.update`: Update of the Arnoldi iterate after each restart cycle.
- `out.err`: Euclidean norm of the error after each restart cycle (if exact solution is provided as input).
- `out.stop_condition`:
- `out.stop_condition`:

For more details and examples on how to use the code, see also the different demo files `demo_*.m` provided with our code.

When using the code or referring to it, please cite the paper [1].

REFERENCES

[1] A. Frommer, S. Güttel, and M. Schweitzer, *Efficient and stable Arnoldi restarts for matrix functions based on quadrature*, IMACM preprint, (2013).