1 Updates/Upgrades Summary

If you are a FEAST’s first time user, you can skip this section.

Here is a summary of the most important updates/upgrades.

1.1 From v2.1 to v3.0

- A variety of new features have been added in v3.0. This includes support for non-Hermitian matrices, elliptical contours and custom user-defined contours, stochastic estimates for the number of eigenvalues inside search interval, and different quadrature rules. Many new routines have been added. See Table 1 for a summary.

<table>
<thead>
<tr>
<th>Family of Eigenvalue Problems</th>
<th>Routines in v3.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>( AX = BX A ) ( A^H \hat{X} = B^H \hat{X} \Lambda^* )</td>
<td>Elliptical Contours (Standard)</td>
</tr>
<tr>
<td>Real and Symmetric ( A = A^T, B = B^{spd}, X = \hat{X} ) real, ( \Lambda ) real</td>
<td>{s,d}feast_src</td>
</tr>
<tr>
<td>Complex and Hermitian ( A = A^H, B = B^{hpd}, X = \hat{X} ) complex, ( \Lambda ) real</td>
<td>{c,z}feast_hrci</td>
</tr>
<tr>
<td>Complex and Symmetric ( A = A^T, B = B^{hpd}, X = \hat{X} ) complex, ( \Lambda ) complex</td>
<td>{c,z}feast_src</td>
</tr>
<tr>
<td>Real and Non-Symmetric ( A, B ) general, ( X \neq \hat{X} ) complex, ( \Lambda ) complex</td>
<td>{s,d}feast_grci</td>
</tr>
<tr>
<td>Complex and General ( A, B ) general, ( X \neq \hat{X} ) complex, ( \Lambda ) complex</td>
<td>{c,z}feast_grc</td>
</tr>
</tbody>
</table>

Table 1: Summary of all routines in FEAST v3.0 (140 total) - new routines in red

- Non-Hermitian routines use a different variant of the FEAST algorithm than Hermitian cases. The major difference is the use of dual subspaces, \( Q \) and \( \hat{Q} \), corresponding to Right \( \hat{X} \) and Left \( \hat{X} \) eigenvectors. Also, the search interval must become 2-dimensional to account for complex eigenvalues. More detail is given in:

  FEAST Eigensolver for Non-Hermitian Problems,

- FEAST now offers multiple quadrature rules: Gauss, Trapezoidal and Zolotarev (for the Hermitian case), as well as elliptical complex contour. More detail is given in:

  Optimized Quadrature Rules and Load Balancing for the FEAST Eigenvalue Solver,
  S. Gütteil, E. Polizzi, P. T. Tang, G. Viaud, \url{http://arxiv.org/abs/1407.8078}

- All FEAST routines can be called within their “expert mode” version which features new user input lists for nodes and weights.

- Stochastic estimates for the number of eigenvalues inside of the search interval are now available. This feature can help users in estimating a value for the search subspace \( M_0 \). Refer to the following publication for more information.

  Efficient Estimation of Eigenvalue Counts in an Interval,
  E. Di Napoli, E. Polizzi, Y. Saad, \url{http://arxiv.org/abs/1308.4275}

- Various utility routines have also been added (see section 6). We note in particular the possibility for the users to design their own contour shape in the complex plane. This is particularly helpful for non-Hermitian routines as it grants flexibility in targeting specific eigenvalues. See Section 6 for additional information.
FEAST PARAMETERS- new or updated fpm parameters:

- \text{fpm(2)} is updated- includes more options for #nodes in the half-contour (for Hermitian FEAST)
  
  If \text{fpm(16)}=0,2, values permitted [1 to 20, 24, 32, 40, 48, 56]
  
  If \text{fpm(16)}=1, all values permitted

- \text{fpm(6)} is updated- default value changed to 1 -
  
  Convergence criteria on trace (0) or eigenvectors relative residual (1)

- \text{fpm(8)} is added - Total number of contour integration nodes (i.e. complex shifts) for non-Hermitian FEAST.
  
  If \text{fpm(17)}=0, values permitted [2 to 40, 48, 64, 80, 96, 112]
  
  If \text{fpm(17)}=1, all values permitted
  
  Remark: \text{fpm(8)} represents the #nodes for the full contour while \text{fpm(2)} represents the #nodes for the half-contour used by Hermitian FEAST.

- \text{fpm(10)} is added - can be used with the FEAST predefined driver interfaces (0: default, 1: store all the linear system factorizations).
  
  Remark: (i) storing the factorizations will significantly improve the performances (for FEAST DENSE in particular), but can significantly increase the memory usage; (ii) option 1 works with FEAST-MPI as well - store all factors associated to a given mpi process

- \text{fpm(14)} is modified - include option 2
  
  0: default normal FEAST execution
  
  1: return only subspace Q size M0 after 1 contour
  
  2: return stochastic estimates of the #eigenvalue (in argument 'M' and 'res' for running average)

- \text{fpm(16)} is added - Integration type for symmetric (0: Gauss/Default, 1: Trapezoidal, 2: Zolotarev)

- \text{fpm(17)} is added - Integration type for non-symmetric (0: Gauss, 1: Trapezoidal/Default)

- \text{fpm(18)} is added - Ratio for ellipsoid contour - \text{fpm(18)}/100 is ratio 'vertical axis'/'horizontal axis' of the ellipse using the definition of the search contour. For example:
  
  value 100 is the default (circle);
  value 50 will create a 50% flat ellipse;
  value 200 will create a 200% tall ellipse.

- \text{fpm(19)} is added - Rotation angle in degree [-180:180] for Ellipsoid contour and using FEAST non-Hermitian- Origin of the rotation is the vertical axis.