Open TURNS and the HPC

- « H-Matrices methods principles. Applications to OpenTURNS»: T. Abboud (Ecole Polytechnique)

- « The H-Matrices in Open TURNS v1.5 »: D. Barbier (IMACS)

The H-Matrices in Open TURNS v1.5

- H-matrix library developed by Airbus Group Innovations
    - Sequential code available at https://github.com/jeromerobert/hmat-oss/
    - GPLv2 license, with a license exception to link it to OpenTURNS
    - Parallel implementation available under a commercial license

- H-matrix and OpenTURNS:
    - In v1.5, only used by TemporalNormalProcess class to simulate Gaussian fields
    - In v1.6, H-matrix library fully wrapped, also used in kriging

The H-Matrices in OpenTURNS v1.5

- Early stage interface, HMatrix was primarily designed by AGI for their own BEM solvers
- Univariate and multivariate covariance models are supported
- H-matrix internally renumbers degrees of freedom
    - Comparing results between Lapack and Hmatrix may be difficult or impossible
    - Cholesky factors cannot be compared, but we can compare LLt product
    - Following results have been obtained by using an already ordered mesh, but internal ordering of H-Matrix is not available within OpenTURNS

The H-Matrices in Open TURNS v1.5

```
# Only 3D meshes are supported in OpenTURNS 1.5
spatial_dimension = 3
spatial_range = mesh.getVertices().computeRange()
rho = 0.1
covariance_model = AbsoluteExponential(spatial_dimension, rho * spatial_range.norm())
# 0=Lapack, 1=HMat, 2=Gibbs
ResourceMap.SetAsUnsignedInteger("TemporalNormalProcess-SamplingMethod", 0)
normal_process_L = TemporalNormalProcess(covariance_model, mesh)
RandomGenerator.SetSeed(19)
field_L = normal_process_L.getRealization()
```

$$C = L\,L^t \quad Y = LX$$

```
ResourceMap.SetAsUnsignedInteger("TemporalNormalProcess-SamplingMethod", 1)
normal_process_H = TemporalNormalProcess(covariance_model, mesh)
RandomGenerator.SetSeed(19)
field_H = normal_process_H.getRealization()
```

$$C \approx \tilde{L}\,\tilde{L}^t \quad \tilde{Y} = \tilde{L}X$$

```
# Comparison does not work in general because Hmatrix reorders vertices
print((field_L.getValues() - field_H.getValues()).computeRawMoment(2))
```

The H-Matrices in Open TURNS v1.5

Epsilon	LLt (s)	getRealization (s)	Error	Matrix memory ratio
1e-4	12	0.02	1.4e-5	1 / 8.0
1e-7	25	0.03	3.9e-11	1 / 5.0
1e-10	56	0.05	7.2e-17	1 / 3.1
1e-13	99	0.07	1.2e-22	1 / 2.1
1e-16	400	0.11	2.7e-23	1 / 1.5
LAPACK	30	0.06	0	1

Mesh containing 9644 vertices, machine with 8 cores and 8 GB RAM, opensource HMatrix (sequential)

The H-Matrices in Open TURNS v1.5

Epsilon	LLt (s)	getRealization (s)	Error	Matrix memory ratio
1e-4	16	0.05	1.e-4	1 / 14.1
1e-7	24	0.04	1.e-11	1 / 8.3
1e-10	39	0.05	1.e-16	1 / 5.2
1e-13	60	0.06	1.e-22	1 / 3.5
1e-16	299	0.07	1.e-25	1 / 2.3
LAPACK	175	0.15	-	1

Mesh containing 18292 vertices, machine with 12 cores and 48 GB RAM

The H-Matrices in Open TURNS v1.5

Epsilon	LLt (s)	getRealization (s)	Error	Matrix memory ratio
1e-8	452	0.04	-	1 / 28.6
1e-12	853	0.06	-	1 / 15.6
LAPACK	-	-	-	1

Mesh containing 134331 vertices, machine with 12 cores and 48 GB RAM

The H-Matrices in Open TURNS v1.5

H-Matrix resource keys

- CovarianceHMatrix-AdmissibilityFactor
  - Coefficient of Hackbusch admissibility condition (default : 2.0)
- CovarianceHMatrix-CompressionMethod
  - 0 = SVD : best results, but unpractical on large blocks because of high algorithmic complexity
  - 1 = ACA : default
  - 2 = partial ACA : like ACA, but Gauss pivot is looked for only by scanning one line and one row ⇒ much faster than ACA, but may give wrong results
  - 3 = ACA+ : similar to partial ACA, more robust
- CovarianceHMatrix-AssemblyEpsilon
  - Controls low-rank approximation when assembling compressed blocks (default 1.e-4)
- CovarianceHMatrix-RecompressionEpsilon
  - Controls low-rank approximation when recompressing matrices, during factorization (default : 1.e-4)
- CovarianceHMatrix-MaxLeafSize
  - Maximal number of elements in smallest blocks (default : 100)

The H-Matrices in Open TURNS v1.5

Caveats

- Covariance matrix is SPD, but its conditioning number grows dramatically when number of points by correlation length becomes large
  - Depends on covariance model
  - Decrease correlation length if factorization fails
- Approximated covariance matrix not necessarily SPD ⇒ Cholesky factorization can fail
  - Add a nugget factor on diagonal to regularize approximated covariance matrix, controlled by resource keys TemporalNormalProcess-StartingScaling and TemporalNormalProcess-MaximalScaling

The H-Matrices in Open TURNS v1.5

Perspectives

- In v1.6:
  - Allows using H-matrices in kriging ⇒ improved HMatrix encapsulation, resource keys renamed from CovarianceHMatrix-* into HMatrix-*
- Future:
  - More customization, e.g. allows to specify other admissibility conditions, like HODLR (Hierarchical Off-Diagonal Low-Rank) to improve compression and thus complexity
  - Work on automatic regularization: compute a nugget factor which will require less try-fail computations without being excessively large
  - Conditioning number of the covariance matrix is highly dependent on provided mesh; detect overrefined meshes?
  - Any other linear algebra operations in OpenTURNS?