v1.1.0

Search docs

LAPACK Functions
Logging
Markdown
Memory-mapped I/O
Pkg
Printf
Profiling
The Julia REPL
Random Numbers
SHA
Serialization
Shared Arrays
Sockets
Sparse Arrays
Statistics
Unit Testing
UUIDs
Unicode
Developer Documentation
Reflection and introspection
Documentation of Julia's Internals
Initialization of the Julia runtime
Julia ASTs
More about types
Memory layout of Julia Objects
Eval of Julia code
Calling Conventions
High-level Overview of the Native-Code Generation Process
Julia Functions
Base.Cartesian
Talking to the compiler (the :meta mechanism)
SubArrays
isbits Union Optimizations
System Image Building

```
julia> F = svd(A);

julia> F.U * Diagonal(F.S) * F.Vt
4×5 Array{Float64,2}:
 1.0  0.0  0.0  0.0  2.0
 0.0  0.0  3.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0
 0.0  2.0  0.0  0.0  0.0
```

source

```
svd(A, B) -> GeneralizedSVD
```

Compute the generalized SVD of `A` and `B`, returning a `GeneralizedSVD` factorization object `F`, such that `A = F.U*F.D1*F.R0*F.Q'` and `B = F.V*F.D2*F.R0*F.Q'`.

For an M-by-N matrix `A` and P-by-N matrix `B`,

- `U` is a M-by-M orthogonal matrix,
- `V` is a P-by-P orthogonal matrix,
- `Q` is a N-by-N orthogonal matrix,
- `D1` is a M-by-(K+L) diagonal matrix with 1s in the first K entries,
- `D2` is a P-by-(K+L) matrix whose top right L-by-L block is diagonal,
- `R0` is a (K+L)-by-N matrix whose rightmost (K+L)-by-(K+L) block is nonsingular upper block triangular,

`K+L` is the effective numerical rank of the matrix `[A; B]`.

Iterating the decomposition produces the components `U`, `V`, `Q`, `D1`, `D2`, and `R0`.

The entries of `F.D1` and `F.D2` are related, as explained in the LAPACK documentation for the generalized SVD and the xGGSVD3 routine which is called underneath (in LAPACK 3.6.0 and newer).

Examples

```
julia> A = [1. 0.; 0. -1.]
2×2 Array{Float64,2}:
 1.0   0.0
 0.0  -1.0

julia> B = [0. 1.; 1. 0.]
2×2 Array{Float64,2}:
 0.0  1.0
 1.0  0.0

julia> F = svd(A, B);

julia> F.U*F.D1*F.R0*F.Q'
2×2 Array{Float64,2}:
 1.0   0.0
 0.0  -1.0

julia> F.V*F.D2*F.R0*F.Q'
2×2 Array{Float64,2}:
 0.0  1.0
 1.0  0.0
```

source

`LinearAlgebra.svd!` — *Function*.