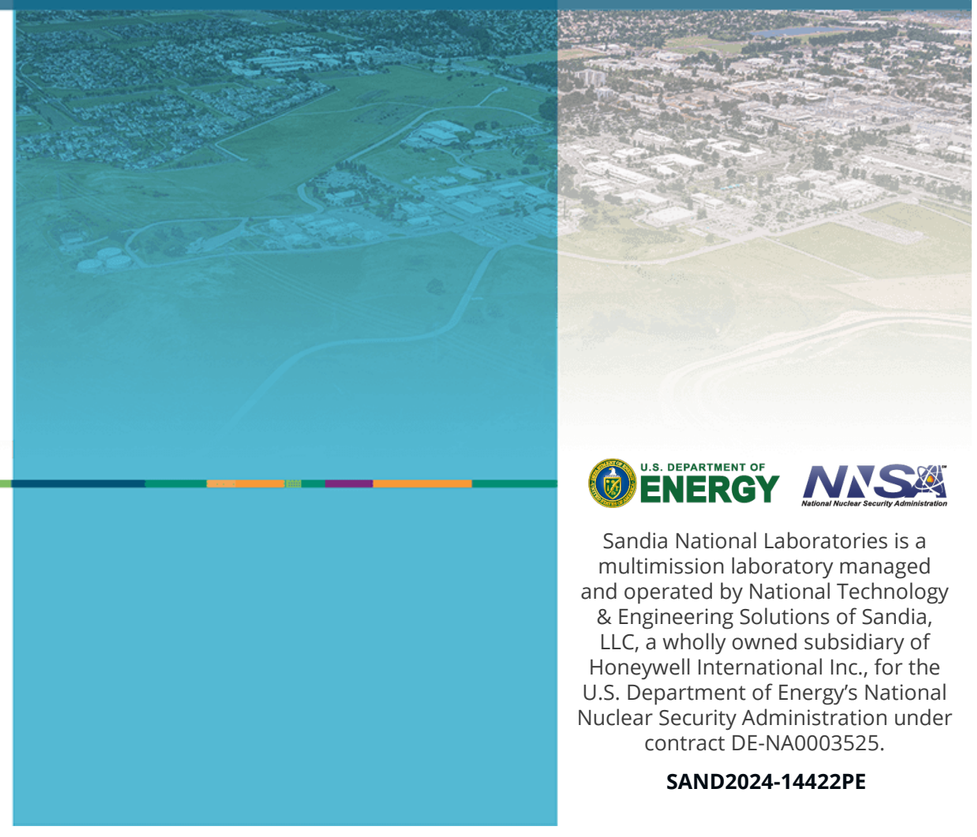
Belos Serial Dense Matrix Abstraction

Heidi Thornquist (1325) and Jennifer Loe (1465)

10/24/2024



Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

SAND2024-14422PE

Belos Templated Linear Solvers

- Next generation linear solver libraries, written in templated C++
  - Iterative methods for solving sparse, matrix-free systems
- Provide a generic interface to a collection of algorithms for solving linear problems
- Algorithms developed with generic programming techniques
  - Algorithmic components:
    - Ease the implementation of complex algorithms
  - Operator/MultiVector interface *(and Teuchos::ScalarTraits)*:
    - Allow the user to leverage their existing software investment
    - Multi-precision solver capability
  - Dense matrix abstract interface (coming soon!)
    - Allows the user to improve performance in advanced architectures
  - Design offers: Interoperability, extensibility, and reusability
- Includes block linear solvers

Belos Solver Categories

Belos provides solvers for:

- Single RHS: $Ax = b$
- Multiple RHS (available simultaneously): $AX = B$
- Multiple RHS (available sequentially): $Ax_i = b_i$, $i=1,\ldots,k$
- Sequential Linear systems: $A_i x_i = b_i$, $i=1,\ldots,k$
- Linear Least Squares: $\min \| Ax - b \|_2$

Leverage research advances of solver community:

- Block methods: block GMRES [Vital], block CG/BICG [O'Leary]
- "Seed" solvers: hybrid GMRES [Nachtigal, et al.]
- "Recycling" solvers for sequences of linear systems: RCG, GCRO-DR [Parks, et al.]
- Restarting, orthogonalization techniques
- Multi-precision methods, communication-avoiding methods

Belos Solvers

Hermitian Systems ($A = A^H$)

- CG / Block CG
- Pseudo-Block CG (Perform single-vector algorithm simultaneously)
- RCG (Recycling Conjugate Gradients)
- PCPG (Projected CG)
- MINRES

Non-Hermitian System ($A \neq A^H$)

- Block GMRES
- Pseudo-Block GMRES (Perform single-vector algorithm simultaneously)
- Block FGMRES (Variable preconditioner)
- Hybrid GMRES
- TFQMR
- GCRO-DR / Block GCRO-DR (Recycling GMRES)
- RCG (Recycling CG)

Linear Least Squares

- LSQR

Belos
(Framework Overview)

$x^{(0)}$ is an initial guess
for $j = 1, 2, ...$
  Solve r from $Mr = b - Ax^{(0)}$
  $v^{(1)} = r/\|r\|_2$
  $s := \|r\|_2 e_1$
  for $i = 1, 2, ..., m$
    Solve w from $Mw = Av^{(i)}$
    for $k = 1, ..., i$
      $h_{k,i} = (w, v^{(k)})$
      $w = w - h_{k,i} v^{(k)}$
    end
    $h_{i+1,i} = \|w\|_2$
    $v^{(i+1)} = w/h_{i+1,i}$
    apply $J_1, ..., J_{i-1}$ on $(h_{1,i}, ..., h_{i+1,i})$
    construct J_i, acting on ith and $(i+1)$st component
    of $h_{.,i}$, such that $(i+1)$st component of $J_i h_{.,i}$ is 0
    $s := J_i s$
    if $s(i+1)$ is small enough then (UPDATE$(\tilde{x}, i)$ and quit)
  end
  UPDATE$(\tilde{x}, m)$
end

GMRES Example

Belos
(Framework Overview)

Multivector Traits

$x^{(0)}$ is an initial guess
for $j = 1, 2, ...$
  Solve r from $Mr = b - Ax^{(0)}$
  $v^{(1)} = r/\|r\|_2$
  $s := \|r\|_2 e_1$
  for $i = 1, 2, ..., m$
    Solve w from $Mw = Av^{(i)}$
    for $k = 1, ..., i$
      $h_{k,i} = (w, v^{(k)})$
      $w = w - h_{k,i} v^{(k)}$
    end
    $h_{i+1,i} = \|w\|_2$
    $v^{(i+1)} = w/h_{i+1,i}$
    apply $J_1, ..., J_{i-1}$ on $(h_{1,i}, ..., h_{i+1,i})$
    construct J_i, acting on ith and $(i+1)$st component
    of $h_{.,i}$, such that $(i+1)$st component of $J_i h_{.,i}$ is 0
    $s := J_i s$
    if $s(i+1)$ is small enough then (UPDATE$(\tilde{x}, i)$ and quit)
  end
  UPDATE$(\tilde{x}, m)$
end

GMRES Example

Belos
(Framework Overview)

Problem Classes / Operator Traits

Multivector Traits

$x^{(0)}$ is an initial guess
for $j = 1, 2, ...$
  Solve r from $Mr = b - Ax^{(0)}$
  $v^{(1)} = r/\|r\|_2$
  $s := \|r\|_2 e_1$
  for $i = 1, 2, ..., m$
    Solve w from $Mw = Av^{(i)}$
    for $k = 1, ..., i$
      $h_{k,i} = (w, v^{(k)})$
      $w = w - h_{k,i} v^{(k)}$
    end
    $h_{i+1,i} = \|w\|_2$
    $v^{(i+1)} = w/h_{i+1,i}$
    apply $J_1, ..., J_{i-1}$ on $(h_{1,i}, ..., h_{i+1,i})$
    construct J_i, acting on ith and $(i+1)$st component
    of $h_{.,i}$, such that $(i+1)$st component of $J_i h_{.,i}$ is 0
    $s := J_i s$
    if $s(i+1)$ is small enough then (UPDATE$(\tilde{x}, i)$ and quit)
  end
  UPDATE$(\tilde{x}, m)$
end

GMRES Example

Belos
(Framework Overview)

Problem Classes / Operator Traits

Multivector Traits

$x^{(0)}$ is an initial guess
for $j = 1, 2, ...$
  Solve r from $Mr = b - Ax^{(0)}$
  $v^{(1)} = r/\|r\|_2$
  $s := \|r\|_2 e_1$
  for $i = 1, 2, ..., m$
    Solve w from $Mw = Av^{(i)}$
    for $k = 1, ..., i$
      $h_{k,i} = (w, v^{(k)})$
      $w = w - h_{k,i} v^{(k)}$
    end
    $h_{i+1,i} = \|w\|_2$
    $v^{(i+1)} = w/h_{i+1,i}$
    apply $J_1, ..., J_{i-1}$ on $(h_{1,i}, ..., h_{i+1,i})$
    construct J_i, acting on ith and $(i+1)$st component
    of $h_{.,i}$, such that $(i+1)$st component of $J_i h_{.,i}$ is 0
    $s := J_i s$
    if $s(i+1)$ is small enough then (UPDATE$(\tilde{x}, i)$ and quit)
  end
  UPDATE$(\tilde{x}, m)$
end

GMRES Example

[Mat]OrthoManager Class
(ICGS, IMGS, DGKS, SVQB, TSQR)

Belos
(Framework Overview)

Problem Classes / Operator Traits

Multivector Traits

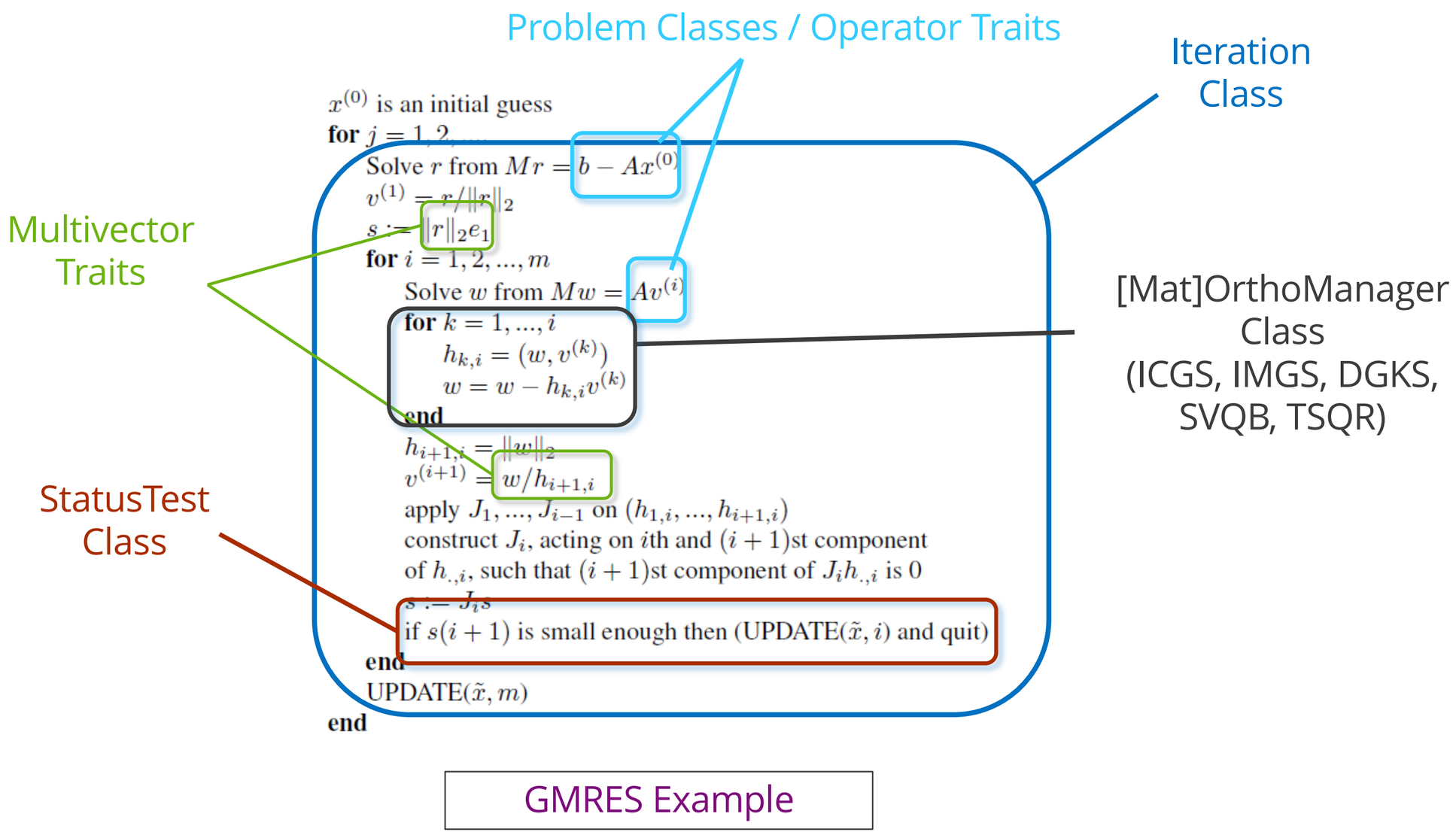
$x^{(0)}$ is an initial guess
for $j = 1, 2, ...$
  Solve r from $Mr = b - Ax^{(0)}$
  $v^{(1)} = r/\|r\|_2$
  $s := \|r\|_2 e_1$
  for $i = 1, 2, ..., m$
    Solve w from $Mw = Av^{(i)}$
    for $k = 1, ..., i$
      $h_{k,i} = (w, v^{(k)})$
      $w = w - h_{k,i}v^{(k)}$
    end
    $h_{i+1,i} = \|w\|_2$
    $v^{(i+1)} = w/h_{i+1,i}$
    apply $J_1, ..., J_{i-1}$ on $(h_{1,i}, ..., h_{i+1,i})$
    construct J_i, acting on ith and $(i+1)$st component
    of $h_{.,i}$, such that $(i+1)$st component of $J_i h_{.,i}$ is 0
    $s := J_i s$
    if $s(i+1)$ is small enough then (UPDATE($\tilde{x}, i$) and quit)
  end
  UPDATE($\tilde{x}, m$)
end

[Mat]OrthoManager Class (ICGS, IMGS, DGKS, SVQB, TSQR)

StatusTest Class

GMRES Example

Belos
(Framework Overview)



GMRES Example

Belos
(Framework Overview)

SolverManager Class

Problem Classes / Operator Traits

Iteration Class

Multivector Traits

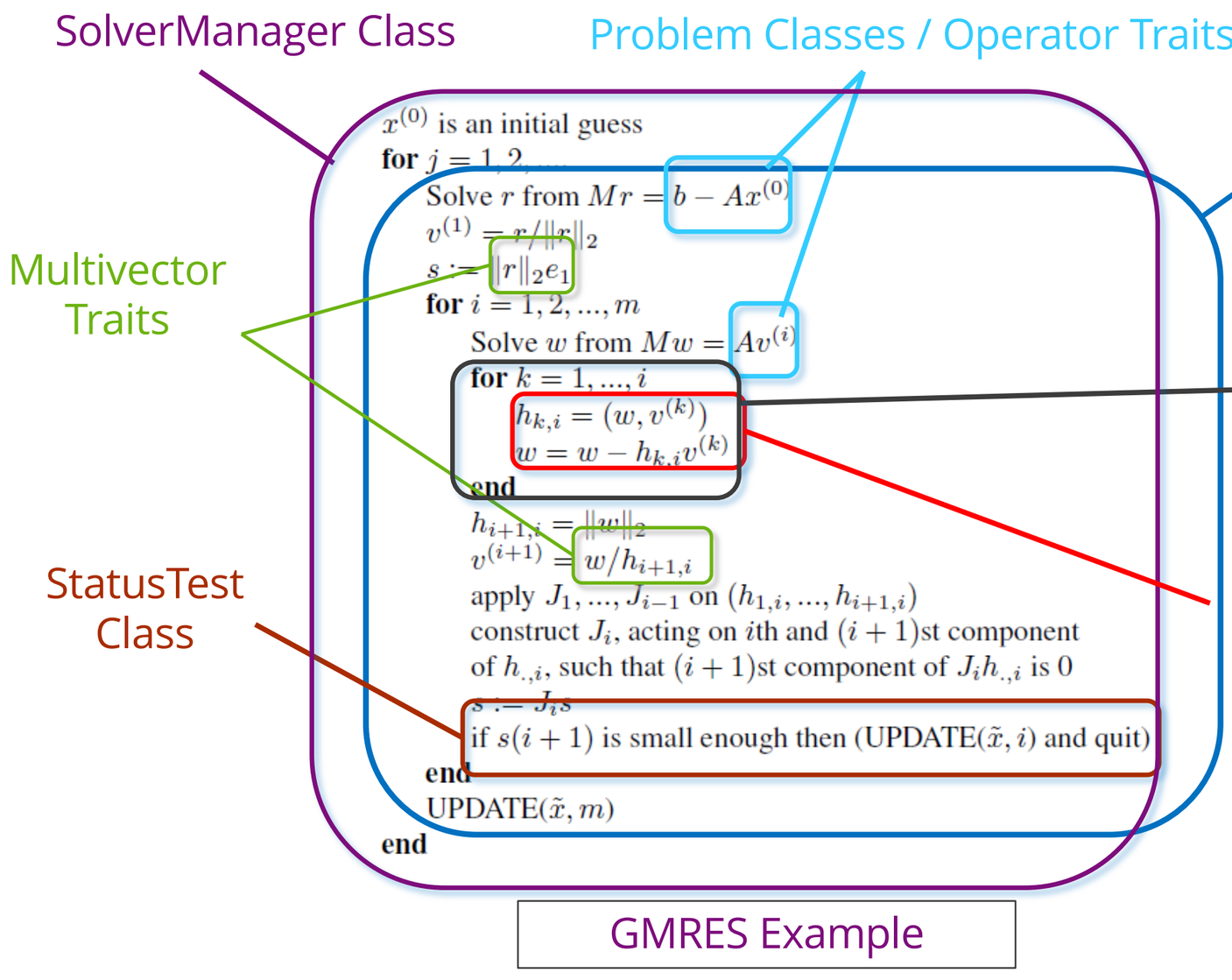
[Mat]OrthoManager Class (ICGS, IMGS, DGKS, SVQB, TSQR)

StatusTest Class

$x^{(0)}$ is an initial guess
for $j = 1, 2, ...$
  Solve r from $Mr = b - Ax^{(0)}$
  $v^{(1)} = r/\|r\|_2$
  $s := \|r\|_2 e_1$
  for $i = 1, 2, ..., m$
    Solve w from $Mw = Av^{(i)}$
    for $k = 1, ..., i$
      $h_{k,i} = (w, v^{(k)})$
      $w = w - h_{k,i}v^{(k)}$
    end
    $h_{i+1,i} = \|w\|_2$
    $v^{(i+1)} = w/h_{i+1,i}$
    apply $J_1, ..., J_{i-1}$ on $(h_{1,i}, ..., h_{i+1,i})$
    construct J_i, acting on ith and $(i+1)$st component
    of $h_{\cdot,i}$, such that $(i+1)$st component of $J_i h_{\cdot,i}$ is 0
    $s := J_i s$
    if $s(i+1)$ is small enough then (UPDATE$(\tilde{x}, i)$ and quit)
  end
  UPDATE$(\tilde{x}, m)$
end

GMRES Example

Belos
(Framework Overview)



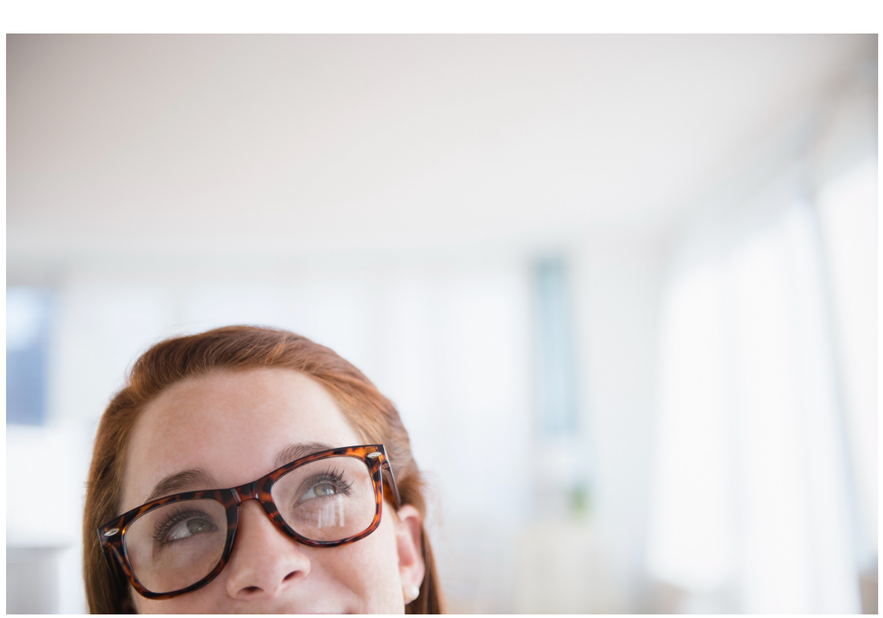
GMRES Example

What is the Dense Matrix Traits interface?

An abstraction of the dense matrix object intended to improve iterative solver performance in advanced architectures

- Default: Teuchos::SerialDenseMatrix<>
- Optionally: Kokkos::DualView<>

So, what is the minimal interface for Belos linear solvers?



What is the Dense Matrix Traits interface?

```
static Teuchos::RCP<DM> Create()
static Teuchos::RCP<DM> Create( const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> CreateCopy(const DM & dm, bool transpose=false)
```

What is the Dense Matrix Traits interface?

```
static Teuchos::RCP<DM> Create()
static Teuchos::RCP<DM> Create( const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> CreateCopy(const DM & dm, bool transpose=false)
static void Reshape( DM& dm, const int numrows, const int numcols, bool initZero = true)
```

What is the Dense Matrix Traits interface?

```
static Teuchos::RCP<DM> Create()
static Teuchos::RCP<DM> Create( const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> CreateCopy(const DM & dm, bool transpose=false)
static void Reshape( DM& dm, const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> Subview( DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static Teuchos::RCP<const DM> SubviewConst( const DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static Teuchos::RCP<DM> SubviewCopy( const DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
```

What is the Dense Matrix Traits interface?

```
static Teuchos::RCP<DM> Create()
static Teuchos::RCP<DM> Create( const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> CreateCopy(const DM & dm, bool transpose=false)
static void Reshape( DM& dm, const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> Subview( DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static Teuchos::RCP<const DM> SubviewConst( const DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static Teuchos::RCP<DM> SubviewCopy( const DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static ScalarType* GetRawHostPtr(DM & dm )
static ScalarType const * GetConstRawHostPtr(const DM & dm )
```

What is the Dense Matrix Traits interface?

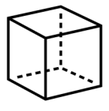
```
static Teuchos::RCP<DM> Create()
static Teuchos::RCP<DM> Create( const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> CreateCopy(const DM & dm, bool transpose=false)
static void Reshape( DM& dm, const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> Subview( DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static Teuchos::RCP<const DM> SubviewConst( const DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static Teuchos::RCP<DM> SubviewCopy( const DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static ScalarType* GetRawHostPtr(DM & dm )
static ScalarType const * GetConstRawHostPtr(const DM & dm )
static ScalarType & Value( DM& dm, const int i, const int j )
static const ScalarType & ValueConst( const DM& dm, const int i, const int j )
```

What is the Dense Matrix Traits interface?

```
static Teuchos::RCP<DM> Create()
static Teuchos::RCP<DM> Create( const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> CreateCopy(const DM & dm, bool transpose=false)
static void Reshape( DM& dm, const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> Subview( DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static Teuchos::RCP<const DM> SubviewConst( const DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static Teuchos::RCP<DM> SubviewCopy( const DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static ScalarType* GetRawHostPtr(DM & dm )
static ScalarType const * GetConstRawHostPtr(const DM & dm )
static ScalarType & Value( DM& dm, const int i, const int j )
static const ScalarType & ValueConst( const DM& dm, const int i, const int j )
static void SyncDeviceToHost(DM & dm)
static void SyncHostToDevice(DM & dm)
```

What is the Dense Matrix Traits interface?

```
static Teuchos::RCP<DM> Create()
static Teuchos::RCP<DM> Create( const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> CreateCopy(const DM & dm, bool transpose=false)
static void Reshape( DM& dm, const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> Subview( DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static Teuchos::RCP<const DM> SubviewConst( const DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static Teuchos::RCP<DM> SubviewCopy( const DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static ScalarType* GetRawHostPtr(DM & dm )
static ScalarType const * GetConstRawHostPtr(const DM & dm )
static ScalarType & Value( DM& dm, const int i, const int j )
static const ScalarType & ValueConst( const DM& dm, const int i, const int j )
static void SyncDeviceToHost(DM & dm)
static void SyncHostToDevice(DM & dm)
static int GetNumRows( const DM& dm )
static int GetNumCols( const DM& dm )
static int GetStride( const DM& dm )
```



What is the Dense Matrix Traits interface?

```
static Teuchos::RCP<DM> Create()
static Teuchos::RCP<DM> Create( const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> CreateCopy(const DM & dm, bool transpose=false)
static void Reshape( DM& dm, const int numrows, const int numcols, bool initZero = true)
static Teuchos::RCP<DM> Subview( DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static Teuchos::RCP<const DM> SubviewConst( const DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static Teuchos::RCP<DM> SubviewCopy( const DM & source, int numRows, int numCols, int startRow=0, int startCol=0)
static ScalarType* GetRawHostPtr(DM & dm )
static ScalarType const * GetConstRawHostPtr(const DM & dm )
static ScalarType & Value( DM& dm, const int i, const int j )
static const ScalarType & ValueConst( const DM& dm, const int i, const int j )
static void SyncDeviceToHost(DM & dm)
static void SyncHostToDevice(DM & dm)
static int GetNumRows( const DM& dm )
static int GetNumCols( const DM& dm )
static int GetStride( const DM& dm )
static void Add( DM& thisDM, const DM& sourceDM)
static void PutScalar( DM& dm, ScalarType value = Teuchos::ScalarTraits<ScalarType>::zero())
static void Scale( DM& dm, ScalarType value)
static void Randomize( DM& dm)
static void Assign( DM& thisDM, const DM& sourceDM)
static typename Teuchos::ScalarTraits<ScalarType>::magnitudeType NormFrobenius( DM& dm)
static typename Teuchos::ScalarTraits<ScalarType>::magnitudeType NormOne( DM& dm)
```

What are the impacts of this change internal to Belos?

Synchronizations are explicitly integrated into Belos solvers and support classes

- Mostly to perform some computation on the dense matrix object
- Sometimes to provide debug information (optional)

* Uses MvDot

Class	# SyncDeviceToHost	Reason
ICGS / IMGS / DGKS	1*	Debugging
BiCGStab / TFQMR	0*	
Block CG	2	Cholesky solve (POTRF/POTRS)
CG / PseudoBlock CG / PseudoBlock Stochastic CG	2*	Extract scalar coefficients
PCPG	6	Extract scalar coefficients (GESVD)
RCG	48*	Recycling subspace (GEMM/GESV/GETRS/SYGV)
Minres	3	Extract scalar coefficients
Block (F)GMRES / PseudoBlock GMRES	3	Update / solve LSQR (ROT/TRSM)
GCRODR	21	Recycling subspace, update / solve LSQR (ROT/TRSM/ GEMM/GETRF/GETRI/GEQRF/UNGQR/GEEV/GESV/GGEVX)
LSQR	2	Debugging

What are the impacts of this change to Belos MV Traits?

The MultiVectorTraits interface has methods with dense matrix/vector object arguments:

```
static void MvTimesMatAddMv( const ScalarType alpha, const MV& A,
                             const DM& B,
                             const ScalarType beta, MV& mv )

static void MvTransMv( const ScalarType alpha, const MV& A, const MV& mv, DM& B)
```

What are the impacts of this change to Belos MV Traits?

The MultiVectorTraits interface has methods with dense matrix/vector object arguments:

```
static void MvTimesMatAddMv( const ScalarType alpha, const MV& A,
                             const DM& B,
                             const ScalarType beta, MV& mv )
```

```
static void MvTransMv( const ScalarType alpha, const MV& A, const MV& mv, DM& B)
```

What about this one?

```
static void MvDot ( const MV& mv, const MV& A, std::vector<ScalarType> &b)
```

Will a synchronization be hidden within this call?

```
Teuchos::ArrayView<ScalarType> av (b);
mv.dot (A, av (0, numVecs));
```

What are the impacts of this change external to Belos?

All of the classes that directly or indirectly use a dense matrix object will have an additional template argument

```
template<class ScalarType, class MV, class OP, class DM = Teuchos::SerialDenseMatrix<int,ScalarType>>
```

What are the impacts of this change external to Belos?

All of the classes that directly or indirectly use a dense matrix object will have an additional template argument

```
template<class ScalarType, class MV, class OP, class DM = Teuchos::SerialDenseMatrix<int,ScalarType>>
```

Since the default template argument is Teuchos::SerialDenseMatrix<>, this change should not be substantially impactful … initially

What are the impacts of this change external to Belos?

All of the classes that directly or indirectly use a dense matrix object will have an additional template argument

```
template<class ScalarType, class MV, class OP, class DM = Teuchos::SerialDenseMatrix<int,ScalarType>>
```

Since the default template argument is Teuchos::SerialDenseMatrix<>, this change should not be substantially impactful … initially

Partial template specializations of Belos classes may need to be modified

- Stokhos, MueLu ☑

What are the impacts of this change external to Belos?

All of the classes that directly or indirectly use a dense matrix object will have an additional template argument

```
template<class ScalarType, class MV, class OP, class DM = Teuchos::SerialDenseMatrix<int,ScalarType>>
```

Since the default template argument is Teuchos::SerialDenseMatrix<>, this change should not be substantially impactful … initially

Partial template specializations of Belos classes may need to be modified

- Stokhos, MueLu ☑

Direct use of Belos package appears easiest way to employ this new capability

- Thyra interfaces used by Stratimikos may not be compatible with Kokkos::DualView?

What is next for the Dense Matrix Abstraction?

Development and integration of this change has been completed on a fork of Trilinos

- Detailed testing of abstract interface and implementations ☑
- Stokhos, MueLu ☑

What is next for the Dense Matrix Abstraction?

Development and integration of this change has been completed on a fork of Trilinos

- Detailed testing of abstract interface and implementations ☑
- Stokhos, MueLu ☑

Finalizing the refactor

- Fix remaining testing issues
- Obtain initial GPU performance results

What is next for the Dense Matrix Abstraction?

Development and integration of this change has been completed on a fork of Trilinos

- Detailed testing of abstract interface and implementations ☑
- Stokhos, MueLu ☑

Finalizing the refactor

- Fix remaining testing issues
- Obtain initial GPU performance results

Future directions

- Replace synchronizations + BLAS/LAPACK methods with KokkosKernels methods

What is next for the Dense Matrix Abstraction?

Development and integration of this change has been completed on a fork of Trilinos

- Detailed testing of abstract interface and implementations ☑
- Stokhos, MueLu ☑

Finalizing the refactor

- Fix remaining testing issues
- Obtain initial GPU performance results

Future directions

- Replace synchronizations + BLAS/LAPACK methods with KokkosKernels methods

Comments / Questions ?