Belos:

Next-Generation Iterative Solvers

2009 Trilinos User Group Meeting
November 4, 2009

Chris Baker
David Day
Mike Heroux
Mike Parks
Heidi Thornquist (Lead)

SAND 2009-7105P

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company,
for the United States Department of Energy under contract DE-AC04-94AL85000.

Sandia National Laboratories

Outline

- What is Belos?
  - Solvers
  - Belos: What’s in a name?
  - Solver framework structure
  - Simple example
- Spotlight on “Recycling” Solvers
  - Why recycle?
  - Examples
  - Structure of recycling solver
- Summary

What is Belos?

- Solve Ax=b where A large, sparse. Matrix-free.
- Next-generation linear solver library (templated C++)
- Provide generic solver framework solution of large-scale linear systems
- Belos provides solvers for:
  - Single RHS: $Ax = b$
  - Multiple RHS (available simultaneously): $AX = B$
  - Multiple RHS (available sequentially): $Ax_i = b_i$, $i=1,\dots,k$
  - Sequential Linear systems: $A_i x_i = b_i$, $i=1,\dots,k$
- 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 [Parks, et al.]
  - Restarting, orthogonalization techiques
- Belos solver components are: interoperable, extensible, reusable
- **Block linear solvers → Better multicore performance**
- **Multiprecision capability (via Tpetra)**

Sandia National Laboratories

Solvers

- Hermitian Systems ($A = A^H$)
  - Block CG
  - Pseudo-Block CG (Perform single-vector algorithm simultaneously)
  - RCG (Recycling Conjugate Gradients) New!
  - PCPG (Projected CG)

- Non-Hermitian System ($A \neq A^H$)
  - Block GMRES
  - Pseudo-Block GMRES (Perform single-vector algorithm simultaneously)
  - Block FGMRES (Variable preconditioner)
  - Hybrid GMRES
  - TFQMR New!
  - GCRODR (Recycling GMRES)

Sandia National Laboratories

Belos: What's in a name?

Google

belos

Advanced Search
Language Tools

Google Search | I'm Feeling Lucky

Belus - Wikipedia, the free encyclopedia
Belus in Latin or **Belos** in Greek transliteration is one of **...** Belus (Babylonian): the Greek Zeus **Belos** and Latin Jupiter Belus as translations of the **...**
en.wikipedia.org/wiki/Belus - Cached - Similar

WIKIPEDIA
The Free Encyclopedia

article | discussion | edit this page | history

Belus

From Wikipedia, the free encyclopedia

Belus in Latin or **Belos** in Greek transliteration is one of

People and deities

- Ba'al: a title ("lord") in northwest Semitic languages, often applied to particular gods.
- Bel: a title ("lord") in Akkadian, especially applied to the Babylonian god Marduk but
- Belus (Babylonian): the Greek *Zeus Belos* and Latin *Jupiter Belus* as translations of
- Belus (Assyrian): an ancient king of Assyria in classical, legendary history on the ed
- Belus (Egyptian) (sometimes called Belus I): in Greek mythology the son of Poseido
- Belus II. Father of Dido, Pygmalion of Tyre, and Anna. Perhaps identical to the histor
- Belus (Lydian). See Omphale. He was a grandson of Heracles and ancestor of the H

navigation
- Main page
- Contents
- Featured content
- Current events
- Random article

search

Go | Search

WIKIPEDIA
The Free Encyclopedia

article | discussion | edit this page | history

Baal

From Wikipedia, the free encyclopedia

This article **does not cite any references or sources**. Please help improve this article by adding citations to reliable sources. U
removed. *(September 2008)*

For other uses, see Baal (disambiguation).

Ba'al (Hebrew: בעל, pronounced ['baʕal])(ordinarily spelled **Baal** in English) is a Northwest Semitic title and honorific meaning "master" or "lord" that is used for various gods who were patrons of cities in the Levant, cognate to Akkadian *Bēlu*. A **Baalist** or **Baalite** means a worshipper of Baal.

navigation
- Main page
- Contents
- Featured content
- Current events

Ba'al Zebûb

Main article: Beelzebub

Another version of the demon Baal is Beelzebub, or more accurately Ba'al Zebûb or Ba'al Zᵊbûb (Hebrew בעל-זבוב, Ba'al zvuv), who was originally the name of a deity worshipped in the Philistine city of Ekron. Ba'al Zebûb might mean 'Lord of Zebûb', referring to an unknown place named Zebûb, a pun with 'Lord of flies', *zebûb* being a Hebrew collective noun meaning 'fly'. This may mean that the Hebrews were derogating the god of their enemy. Later, Christian writings referred to Ba'al Zebûb as a demon or devil, often interchanged with **Beelzebub**. Either form may appear as an alternate name for Satan or

Let's just stick to the linear algebra…

Sandia National Laboratories

Belos Structure

$x^{(0)}$ 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

Belos Structure

SolverManager 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 $h_{.,i}$, 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

Belos Structure

SolverManager Class

Iteration 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 $h_{.,i}$, 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

Belos Structure

SolverManager Class

LinearProblem, Operator Classes

Iteration Class

$x^{(0)}$ is an initial guess
for $j = 1, 2, \ldots$
  Solve r from $Mr = \boxed{b - Ax^{(0)}}$
  $v^{(1)} = r / \|r\|_2$
  $s := \|r\|_2 e_1$
  for $i = 1, 2, \ldots, m$
    Solve w from $Mw = \boxed{Av^{(i)}}$
    for $k = 1, \ldots, i$
      $h_{k,i} = (w, v^{(k)})$
      $w = w - h_{k,i} v^{(k)}$
    end
    $h_{i+1,i} = \|w\|_2$
    $v^{(i+1)} = w / h_{i+1,i}$
    apply $J_1, \ldots, J_{i-1}$ on $(h_{1,i}, \ldots, h_{i+1,i})$
    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

Belos Structure

SolverManager Class

LinearProblem, Operator Classes

Iteration Class

$x^{(0)}$ is an initial guess
for $j = 1, 2, \ldots$
  Solve r from $Mr = \boxed{b - Ax^{(0)}}$
  $v^{(1)} = r/\|r\|_2$
  $s := \|r\|_2 e_1$
  for $i = 1, 2, \ldots, m$
    Solve w from $Mw = \boxed{Av^{(i)}}$
    for $k = 1, \ldots, i$
      $h_{k,i} = (w, v^{(k)})$
      $w = w - h_{k,i} v^{(k)}$
    end
    $h_{i+1,i} = \|w\|_2$
    $v^{(i+1)} = w/h_{i+1,i}$
    apply $J_1, \ldots, J_{i-1}$ on $(h_{1,i}, \ldots, h_{i+1,i})$
    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

OrthoManager Class

(ICGS, IMGS, DGKS)

Belos Structure

SolverManager Class

LinearProblem, Operator Classes

Iteration Class

OrthoManager Class

(ICGS, IMGS, DGKS)

StatusTest Class

$x^{(0)}$ is an initial guess
for $j = 1, 2, \ldots$
  Solve r from $Mr = \boxed{b - Ax^{(0)}}$
  $v^{(1)} = r/\|r\|_2$
  $s := \|r\|_2 e_1$
  for $i = 1, 2, \ldots, m$
    Solve w from $Mw = \boxed{Av^{(i)}}$
    for $k = 1, \ldots, i$
      $h_{k,i} = (w, v^{(k)})$
      $w = w - h_{k,i} v^{(k)}$
    end
    $h_{i+1,i} = \|w\|_2$
    $v^{(i+1)} = w/h_{i+1,i}$
    apply $J_1, \ldots, J_{i-1}$ on $(h_{1,i}, \ldots, h_{i+1,i})$
    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

Belos Structure

SolverManager Class

LinearProblem, Operator Classes

Iteration Class

$x^{(0)}$ is an initial guess
for $j = 1, 2, \ldots,$
  Solve r from $Mr = \boxed{b - Ax^{(0)}}$
  $v^{(1)} = r/\|r\|_2$
  $s := \|r\|_2 e_1$
  for $i = 1, 2, \ldots, m$
    Solve w from $Mw = \boxed{Av^{(i)}}$
    for $k = 1, \ldots, i$
      $h_{k,i} = (w, v^{(k)})$
      $w = w - h_{k,i} v^{(k)}$
    end
    $h_{i+1,i} = \|w\|_2$
    $v^{(i+1)} = w/h_{i+1,i}$
    apply $J_1, \ldots, J_{i-1}$ on $(h_{1,i}, \ldots, h_{i+1,i})$
    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

OrthoManager Class

(ICGS, IMGS, DGKS)

StatusTest Class

OutputManager Class

Sandia National Laboratories

Example (Step #1 – Initialize System)

```
int main(int argc, char *argv[]) {
  MPI_Init(&argc,&argv);
  Epetra_MpiComm Comm(MPI_COMM_WORLD);
  int MyPID = Comm.MyPID();

  typedef double                            ST;
  typedef Teuchos::ScalarTraits<ST>        SCT;
  typedef SCT::magnitudeType                MT;
  typedef Epetra_MultiVector                MV;
  typedef Epetra_Operator                   OP;
  typedef Belos::MultiVecTraits<ST,MV>     MVT;
  typedef Belos::OperatorTraits<ST,MV,OP>  OPT;

  using Teuchos::ParameterList;
  using Teuchos::RCP;
  using Teuchos::rcp;

  // Get the problem
  std::string filename("orsirr1.hb");
  RCP<Epetra_Map> Map;
  RCP<Epetra_CrsMatrix> A;
  RCP<Epetra_MultiVector> B, X;
  RCP<Epetra_Vector> vecB, vecX;
  EpetraExt::readEpetraLinearSystem(filename, Comm, &A, &Map, &vecX, &vecB);
  X = Teuchos::rcp_implicit_cast<Epetra_MultiVector>(vecX);
  B = Teuchos::rcp_implicit_cast<Epetra_MultiVector>(vecB);
```

Parameters for Templates

Get linear system from disk

`Trilinos/packages/belos/epetra/example/BlockGmres/BlockGmresEpetraExFile.cpp`

Example (Step #2 – Solver Params)

```
bool verbose = false, debug = false, proc_verbose = false;
int frequency = -1;          // frequency of status test output.
int blocksize = 1;           // blocksize
int numrhs = 1;              // number of right-hand sides to solve for
int maxiters = 100;          // maximum number of iterations allowed
int maxsubspace = 50;        // maximum number of blocks
int maxrestarts = 15;        // number of restarts allowed
MT tol = 1.0e-5;             // relative residual tolerance

const int NumGlobalElements = B->GlobalLength();

ParameterList belosList;
belosList.set( "Num Blocks", maxsubspace);              // Maximum number of blocks in Krylov
  factorization
belosList.set( "Block Size", blocksize );               // Blocksize to be used by iterative solver
belosList.set( "Maximum Iterations", maxiters );        // Maximum number of iterations allowed
belosList.set( "Maximum Restarts", maxrestarts );       // Maximum number of restarts allowed
belosList.set( "Convergence Tolerance", tol );          // Relative convergence tolerance requested
int verbosity = Belos::Errors + Belos::Warnings;
if (verbose) {
  verbosity += Belos::TimingDetails + Belos::StatusTestDetails;
  if (frequency > 0)
    belosList.set( "Output Frequency", frequency );
}
if (debug) {
  verbosity += Belos::Debug;
}
belosList.set( "Verbosity", verbosity );
```

Solver Parameters

ParameterList for SolverManager

Trilinos/packages/belos/epetra/example/BlockGmres/BlockGmresEpetraExFile.cpp

Sandia National Laboratories

Example (Step #3 – Solve)

```
// Construct linear problem instance.
Belos::LinearProblem<double,MV,OP> problem( A, X, B );
bool set = problem.setProblem();
if (set == false) {
  std::cout << std::endl << "ERROR:  Belos::LinearProblem failed to
  set up correctly!" << std::endl;
  return -1;
}
```

LinearProblem Object

Template Parameters

SolverManager Object

```
// Start block GMRES iteration
Belos::OutputManager<double> My_OM();
// Create solver manager.
RCP< Belos::SolverManager<double,MV,OP> > newSolver =
  rcp( new Belos::BlockGmresSolMgr<double,MV,OP>(rcp(&problem,false), rcp(&belosList,false)));
// Solve
Belos::ReturnType ret = newSolver->solve();
if (ret!=Belos::Converged) {
  std::cout << std::endl << "ERROR:  Belos did not converge!" << std::endl;
  return -1;
}
std::cout << std::endl << "SUCCESS:  Belos converged!" << std::endl;
return 0;
```

Trilinos/packages/belos/epetra/example/BlockGmres/BlockGmresEpetraExFile.cpp

Spotlight on Recycling

Sequences of Linear Systems

- Consider sequence of linear systems

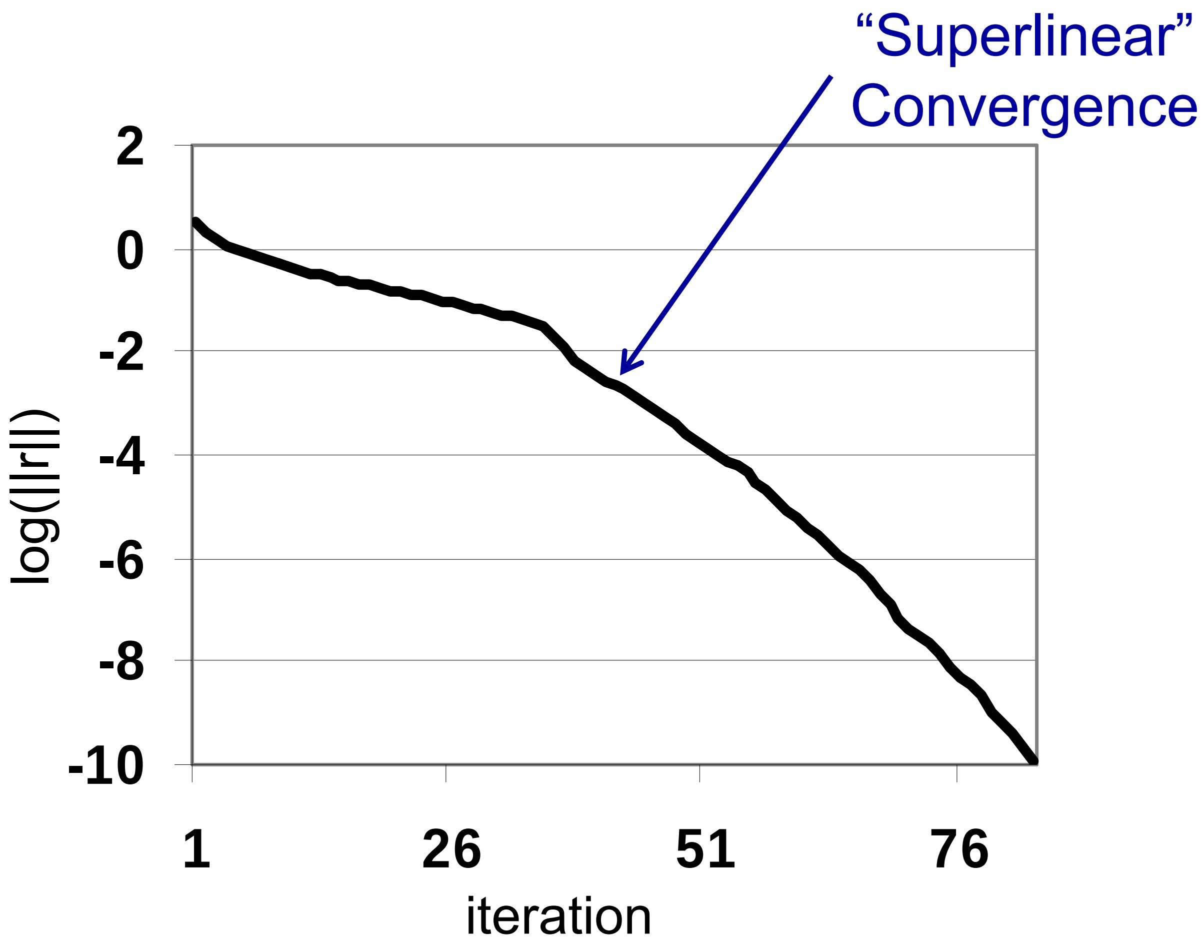
$$\mathbf{A}^{(i)}\mathbf{x}^{(i)} = \mathbf{b}^{(i)} \qquad \mathbf{i=1,2,3,}\ldots$$

- Applications:
  - Newton/Broyden method for nonlinear equations
  - Materials science and computational physics
  - Transient circuit simulation
  - Crack propagation
  - Optical tomography
  - Topology optimization
  - Large-scale fracture in disordered materials
  - Electronic structure calculations
  - Stochastic finite element methods
- Iterative (Krylov) methods build search space and select optimal solution from that space
- **Building search space is dominant cost**
- For sequences of systems, get fast convergence rate and good initial guess immediately by **recycling** selected search spaces from previous systems

Sandia National Laboratories

Why Recycle?

- Typically, dominant subspace exists such that almost any Krylov space (from any starting vector) has large components in that space (why restarting is bad)



Why Recycle?

- Typically, dominant subspace exists such that almost any Krylov space (from any starting vector) has large components in that space (why restarting is bad)
- Optimality derives from orthogonal projection
  - new search directions should be far from this dominant subspace for fast convergence
- If such a dominant subspace persists (approximately) from one system to the next, it can be recycled
  - Typically true when changes to problem are small and/or highly localized

Matrix	Off-the-shelf solver	Recycling Solver	Release
General	GMRES	GCRODR	Trilinos 8
SPD	CG	Recycling CG (RCG)	Trilinos 10
Symmetric Indefinite	MINRES	Recycling MINRES (RMINRES)	N/A

Sandia National Laboratories

Deflation

- Invariant subspace associated with small eigenvalues delays convergence
- Corresponds to smooth modes that change little for small localized changes in the problem

- Remove them to improve convergence!
  - Recycle space = approximate eigenspace

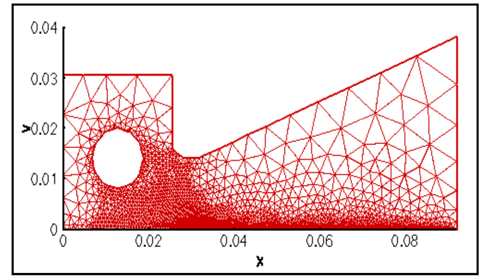
$$\min_{z \in K^m(A, r_0)} \|r_0 - Az\|_2 = \min_{P_m(0)=1} \|p_m(A) r_0\|_2$$

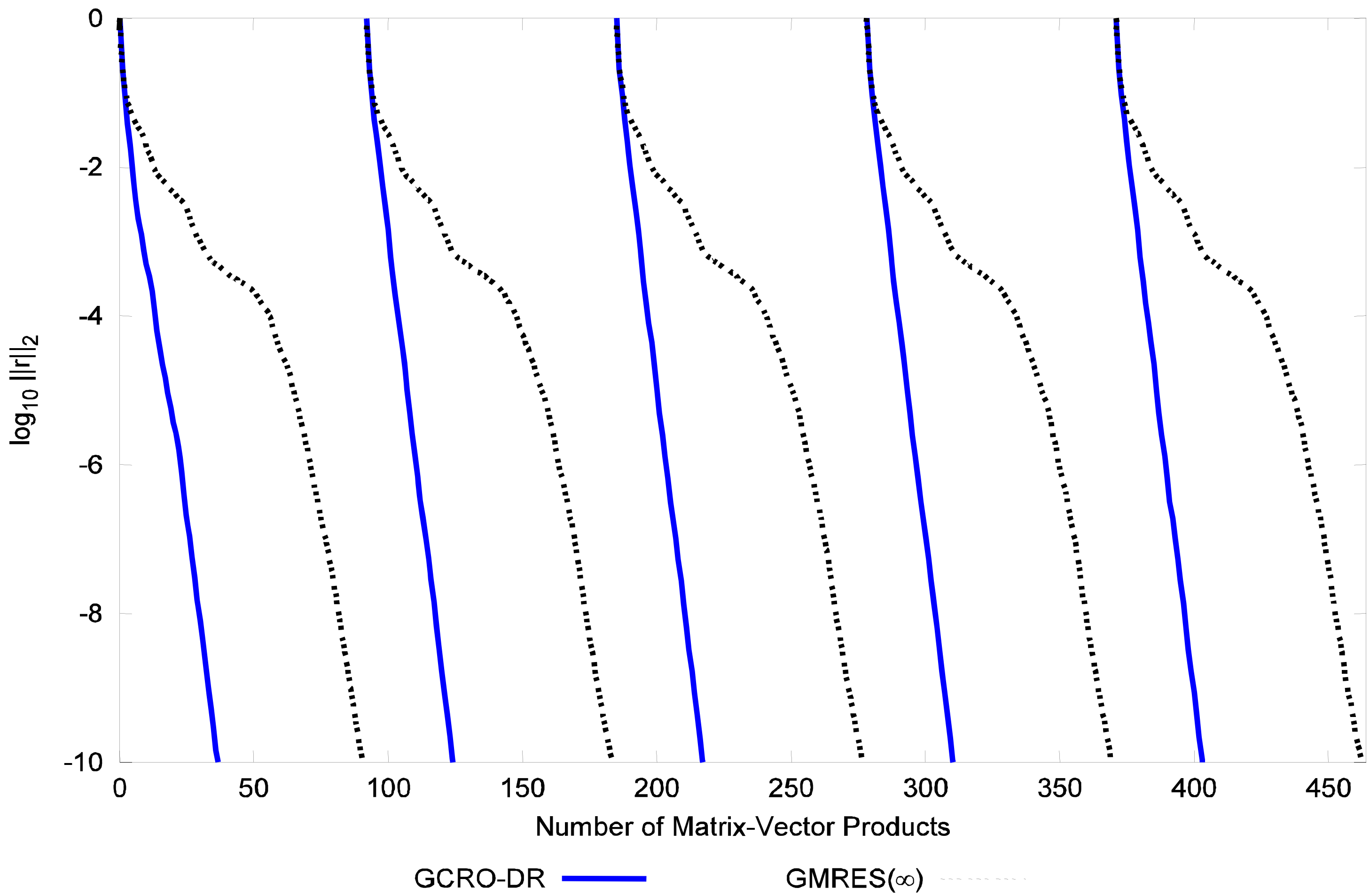
$$\leq \kappa(V) \|r_0\|_2 \min_{P_m(0)=1} \max_{\lambda \in \Lambda(A)} |p_m(\lambda)|$$

- If $\kappa(V)$ is not large (normality assumption) we can improve bound by removing select eigenvalues

Typical Convergence with Recycling

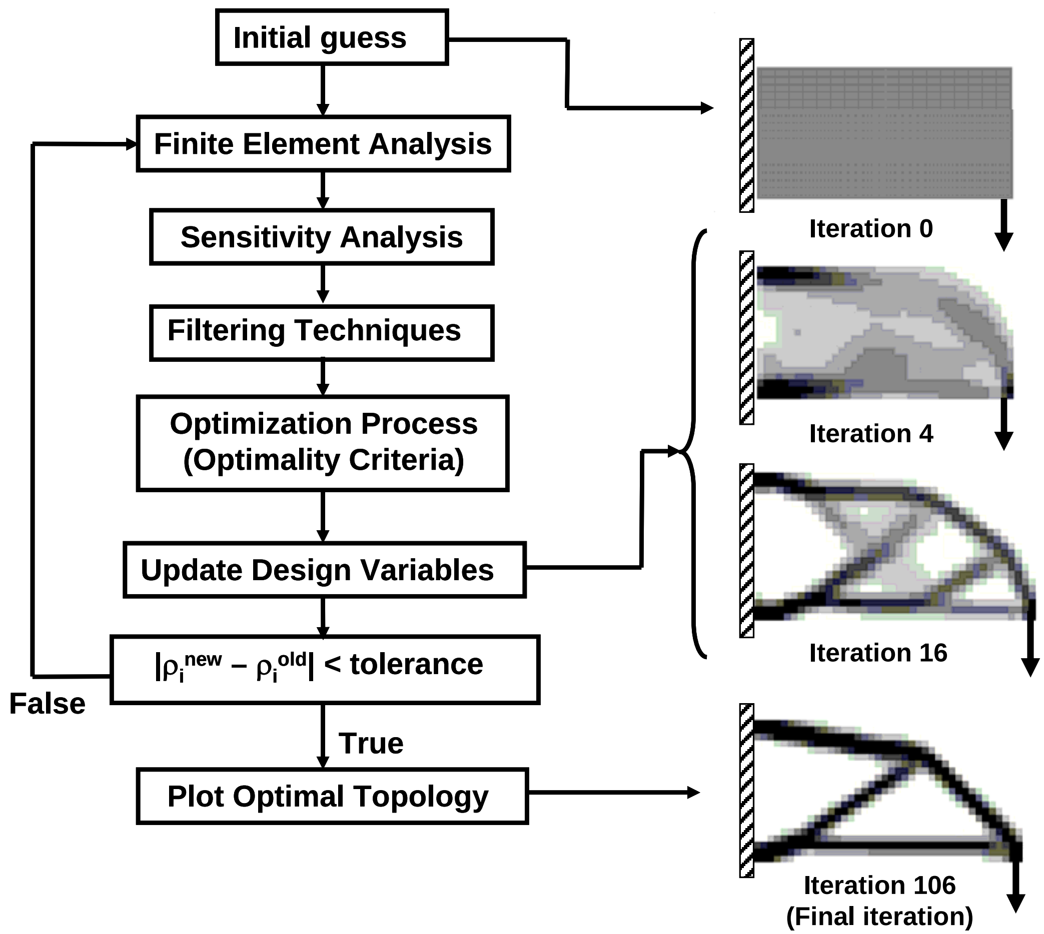
- IC(0) preconditioner
- GMRES – full recurrence
- All Others – Max subspace size 40





Example #1 Topology Optimization*

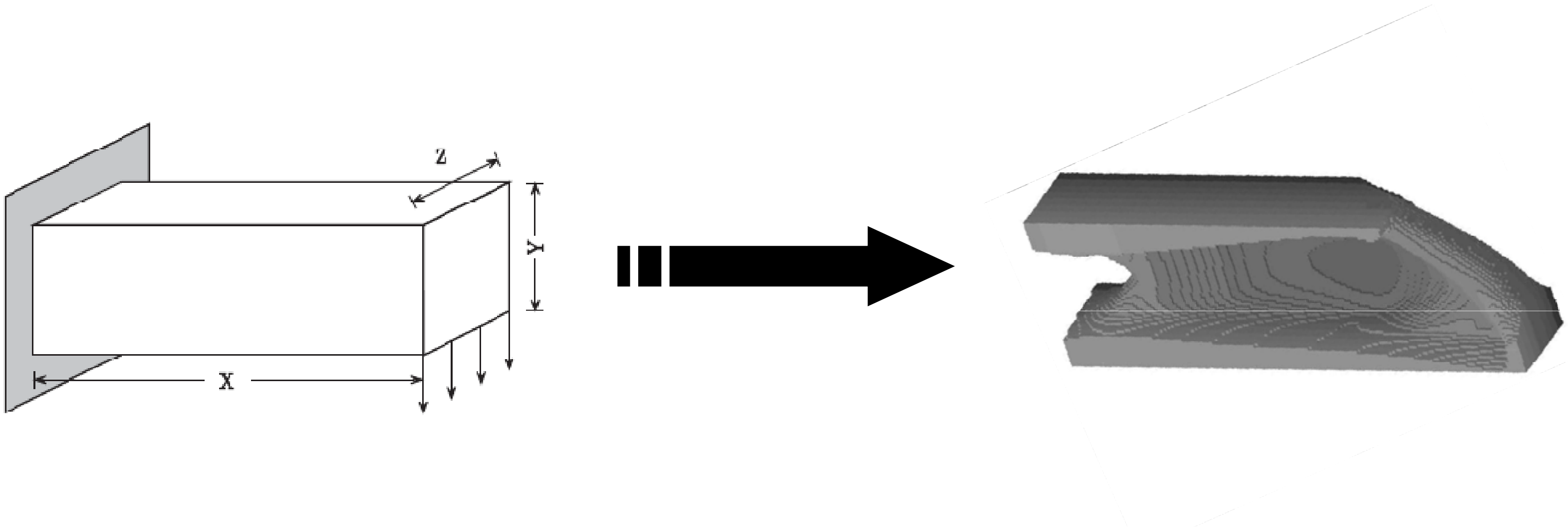
- Optimize material distribution, ρ, in design domain
- Minimize compliance $u^T K(\rho) u$, where $K(\rho)u=f$



*S. Wang, E. de Sturler, and G. H. Paulino, *Large-scale topology optimization using preconditioned Krylov subspace methods with recycling*, International Journal for Numerical Methods in Engineering, Vol. 69, Issue 12, pp. 2441—2468, 2007.

Sandia National Laboratories

Example #1: Topology Optimization



Size	Num. DOFs	Direct Solve Time	Recycling Solve Time
Small	9,360	0.96	1.68
Medium	107,184	179.30	50.41
Large	1,010,160	26154.00	1196.30

Recycling Solve = RMINRES + IC(0) PC

Direct Solve = multifrontal, supernodal Cholesky factorization from TAUCS

*S. Wang, E. de Sturler, and G. H. Paulino, *Large-scale topology optimization using preconditioned Krylov subspace methods with recycling*, International Journal for Numerical Methods in Engineering, Vol. 69, Issue 12, pp. 2441—2468, 2007.

Sandia National Laboratories

Example #2 Stochastic PDEs*

- Stochastic elliptic equation

$$-\nabla \cdot \big(\mathbf{a}(\mathbf{x},\omega)\big)\nabla \mathbf{u}(\mathbf{x},\omega) = \mathbf{f}(\mathbf{x}) \qquad \mathbf{x} \in \mathbf{D}, \omega \in \Omega$$
$$\mathbf{u}(\mathbf{x},\omega) = \mathbf{0} \qquad \mathbf{x} \in \partial\mathbf{D}, \omega \in \Omega$$

- KL expansion + double orthogonal basis + discretization
  - Separate deterministic and stochastic components
  - Yield sequence of uncoupled equations

$$\mathbf{A}^{(i)}\mathbf{x}^{(i)} = \mathbf{b}^{(i)} \qquad \mathbf{i=1,2,3,}\ldots$$

- Preprocess for recycling Krylov solver
  - Use reordering scheme to minimize change in spectra of linear system

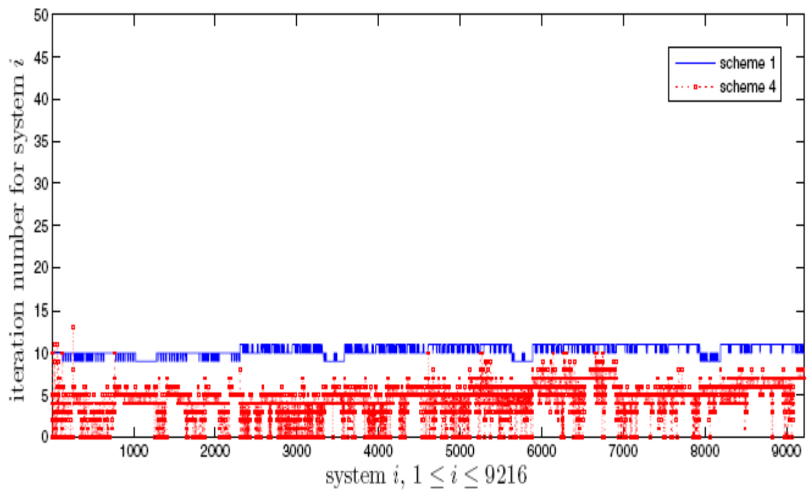
*C. Jin, X-C. Cai, and C. Li, *Parallel Domain Decomposition Methods for Stochastic Elliptic Equations,* SIAM Journal on Scientific Computing, Vol. 29, Issue 5, pp. 2069—2114, 2007.

Sandia National Laboratories

Example #2 Stochastic PDEs*

- Scheme #1: No Krylov recycling
- Scheme #4: Recycle Krylov spaces using reordering
- Many systems require zero iterations!

One-Level ASM



Two-Level ASM

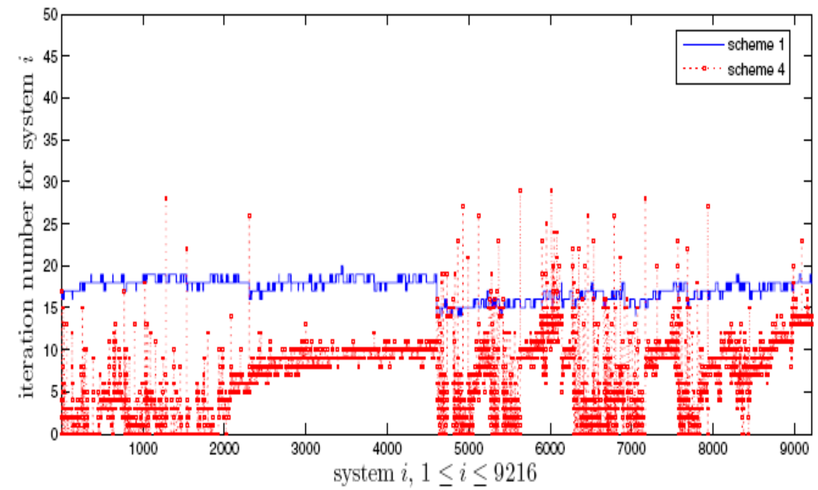


TABLE 4.1

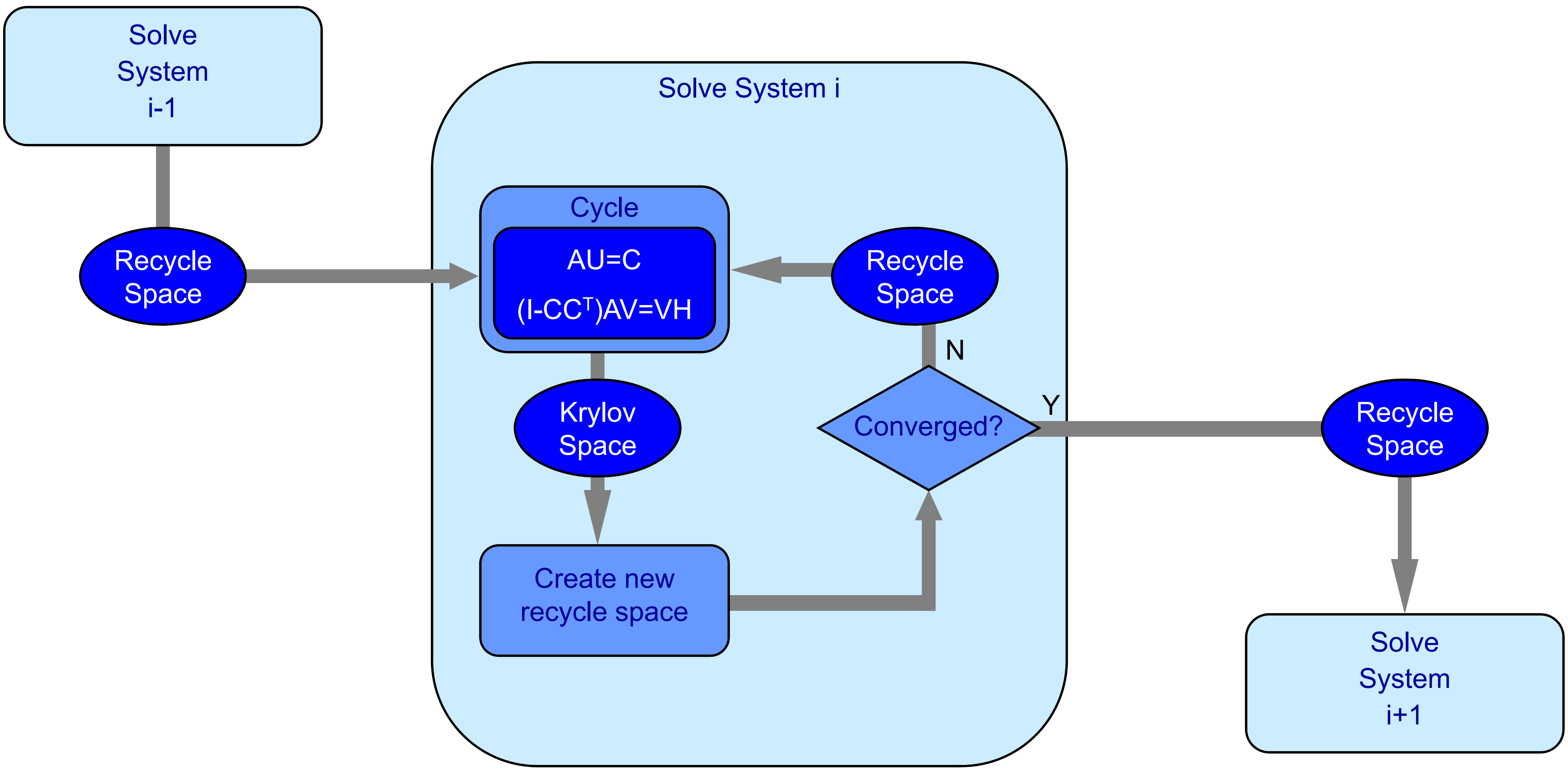
Running time for different schemes and preconditioning (seconds).

Preconditioner	Scheme 1	Scheme 2	Scheme 3	Scheme 4
One-level ASM	12030	7693	4205	3882
Two-level ASM	20980	14130	10740	8476

*C. Jin, X-C. Cai, and C. Li, *Parallel Domain Decomposition Methods for Stochastic Elliptic Equations,* SIAM Journal on Scientific Computing, Vol. 29, Issue 5, pp. 2069—2114, 2007.

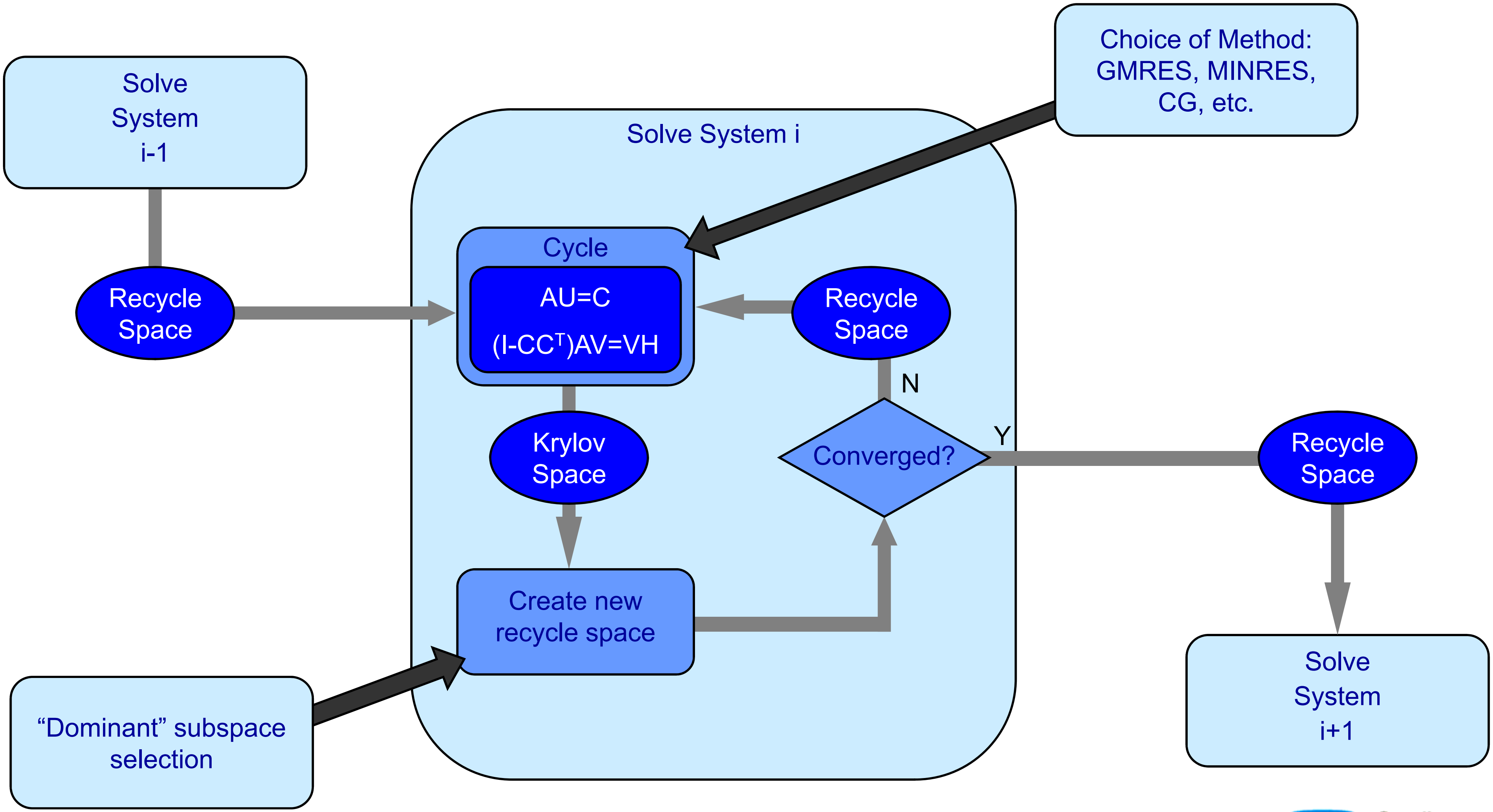
Sandia National Laboratories

Structure of Recycling Solver



Sandia National Laboratories

Structure of Recycling Solver



Summary

- Belos is a **next-generation** linear solver library

- Belos lets you solve:
  - Single RHS: $Ax = b$
  - Multiple RHS (available simultaneously): $AX = B$
  - Multiple RHS (available sequentially): $Ax_i = b_i$, $i=1,\dots,k$
  - Sequential Linear systems: $A_i x_i = b_i$, $i=1,\dots,k$

- Belos contains these solvers:
  - Block CG, Pseudo-Block CG, RCG, PCPG, Block GMRES, Pseudo-Block GMRES, Block FGMRES, Hybrid GMRES, TFQMR, GCRODR

- Check out the Trilinos Tutorial:

  http://trilinos.sandia.gov/Trilinos10.0Tutorial.pdf

- See Belos website for more:

  http://trilinos.sandia.gov/packages/belos