A Linear-Operator Toolbox for Matlab

Michael P. Friedlander
UBC Computer Science

Linear Models

$$Ax \approx b$$

Linear Models

$$Ax \approx b$$

Signal and image processing

- b is observed signal
- x is "true" signal or its representation in some basis/dictionary

Linear Models

$$Ax \approx b$$

Signal and image processing

- b is observed signal
- x is "true" signal or its representation in some basis/dictionary

Linear operators

- may not have a practical matrix representation
- often **fast algorithms** for application of A and A^* (eg, FFT)

Linear Models

$$Ax \approx b$$

Signal and image processing

- b is observed signal
- x is "true" signal or its representation in some basis/dictionary

Linear operators

- may not have a practical matrix representation
- often **fast algorithms** for application of A and A^* (eg, FFT)

Computation

Linear Models

$$Ax \approx b$$

Signal and image processing

- b is observed signal
- x is "true" signal or its representation in some basis/dictionary

Linear operators

- may not have a practical matrix representation
- often **fast algorithms** for application of A and A^* (eg, FFT)

Computation

Ax

Linear Models

$$Ax \approx b$$

Signal and image processing

- b is observed signal
- x is "true" signal or its representation in some basis/dictionary

Linear operators

- may not have a practical matrix representation
- often **fast algorithms** for application of A and A^* (eg, FFT)

Computation

Ax $\qquad$ A^*y

Linear Models

$$Ax \approx b$$

Signal and image processing

- b is observed signal
- x is "true" signal or its representation in some basis/dictionary

Linear operators

- may not have a practical matrix representation
- often **fast algorithms** for application of A and A^* (eg, FFT)

Computation

Ax $\qquad$ A^*y $\qquad$ $A(:, \texttt{idx})x(\texttt{idx})$

Linear Models

$$Ax \approx b$$

Signal and image processing

- b is observed signal
- x is "true" signal or its representation in some basis/dictionary

Linear operators

- may not have a practical matrix representation
- often **fast algorithms** for application of A and A^* (eg, FFT)

Computation

Ax $\qquad$ A^*y $\qquad$ $A(:, \texttt{idx})x(\texttt{idx})$ $\qquad$ $\min \ \frac{1}{2}\|Ax - b\|_2^2 + \rho(x)$

Spot Toolbox

Keeping the "mat" in Matlab

Spot Toolbox

Keeping the "mat" in Matlab

```
A = F'*diag(F*v)*F;
y = A*x;
```

vs

```
function y = circulant(x,v)
  y = ifft(fft(v).*fft(x));
end
y = circulant(x,v);
```

Spot Toolbox

Keeping the "mat" in Matlab

```
A = F'*diag(F*v)*F;
y = A*x;
```

vs

```
function y = circulant(x,v)
  y = ifft(fft(v).*fft(x));
end
y = circulant(x,v);
```

The Toolbox

- free software, GPL

Spot Toolbox

Keeping the "mat" in Matlab

```
A = F'*diag(F*v)*F;
y = A*x;
```

vs

```
function y = circulant(x,v)
  y = ifft(fft(v).*fft(x));
end
y = circulant(x,v);
```

The Toolbox

- free software, GPL
- late-model Matlab required, $\geq$ 7.6 (2008a)

  (extensive use of classdef and packages)

Spot Toolbox

Keeping the "mat" in Matlab

```
A = F'*diag(F*v)*F;
y = A*x;
```

vs

```
function y = circulant(x,v)
  y = ifft(fft(v).*fft(x));
end
y = circulant(x,v);
```

The Toolbox

- free software, GPL
- late-model Matlab required, $\geq$ 7.6 (2008a)

  (extensive use of classdef and packages)
- extensible

Example 1: Image Deblurring

$$b = Ax + r$$

Example 1: Image Deblurring

$$b = Ax + r$$

Spatially-invariant blur

$A = F^* \Lambda F$	$A = D^T \Lambda D$	$A = \mathbf{bttb}(\widehat{c})$
periodic	reflexive	zero

Example 1: Image Deblurring

$$b = Ax + r$$

Spatially-invariant blur

$A = F^* \Lambda F$	$A = D^T \Lambda D$	$A = \mathbf{bttb}(\widehat{c})$
periodic	reflexive	zero

Fast-operators

$$Fx \equiv \sqrt{N}\ \texttt{fft2(x)} \qquad Dx \equiv \texttt{dct2(x)}$$

Example 1: Image Deblurring

$$b = Ax + r$$

Spatially-invariant blur

$A = F^* \Lambda F$	$A = D^T \Lambda D$	$A = \mathbf{bttb}(\widehat{c})$
periodic	reflexive	zero

Fast-operators

$$Fx \equiv \sqrt{N}\ \texttt{fft2(x)} \qquad Dx \equiv \texttt{dct2(x)}$$

Implementing matrix-vector products

$$y = Ax = F^* \Lambda F x \qquad \Longleftrightarrow \qquad y = \texttt{ifft2(fft2(}a_1\texttt{).*fft2(x))}$$

Blurring Demo

Spot Operator Library

Operator type	Matlab function

Spot Operator Library

Operator type	Matlab function
Fast operators	`opConvolve, opCurvelet, opDCT, opDCT2` `opDFT, opDFT2, opDirac, opEye, opHaar, opHaar2,` `opHadamard, opHeaviside, opOnes, opToeplitz` `opToepGauss, opToepSign, opWavelet`

Spot Operator Library

Operator type	Matlab function
Fast operators	`opConvolve, opCurvelet, opDCT, opDCT2 opDFT, opDFT2, opDirac, opEye, opHaar, opHaar2, opHadamard, opHeaviside, opOnes, opToeplitz opToepGauss, opToepSign, opWavelet`
Ensembles	`opBernoulli, opBinary, opGaussian, opSparseBinary`

Spot Operator Library

Operator type	Matlab function
Fast operators	`opConvolve, opCurvelet, opDCT, opDCT2 opDFT, opDFT2, opDirac, opEye, opHaar, opHaar2, opHadamard, opHeaviside, opOnes, opToeplitz opToepGauss, opToepSign, opWavelet`
Ensembles	`opBernoulli, opBinary, opGaussian, opSparseBinary`
Selection	`opExcise, opMask, opRestriction, opSubsref`

Spot Operator Library

Operator type	Matlab function
Fast operators	`opConvolve, opCurvelet, opDCT, opDCT2 opDFT, opDFT2, opDirac, opEye, opHaar, opHaar2, opHadamard, opHeaviside, opOnes, opToeplitz opToepGauss, opToepSign, opWavelet`
Ensembles	`opBernoulli, opBinary, opGaussian, opSparseBinary`
Selection	`opExcise, opMask, opRestriction, opSubsref`
Meta operators	`opBlockDiag, opConj, opDiag, opCTranspose, opKron opDictionary, opZeros, opImage, opInverse, opReal opMinus, opOrthogonal, opPInverse, opPower, opFoG opStack, opSubsAsgn, opSum, opTranspose opEmpty, opZeros`

Spot Operator Library

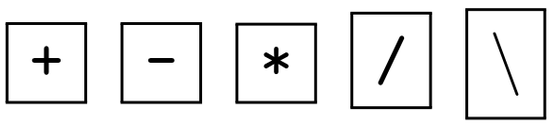
Operator type	Matlab function
Fast operators	`opConvolve, opCurvelet, opDCT, opDCT2 opDFT, opDFT2, opDirac, opEye, opHaar, opHaar2, opHadamard, opHeaviside, opOnes, opToeplitz opToepGauss, opToepSign, opWavelet`
Ensembles	`opBernoulli, opBinary, opGaussian, opSparseBinary`
Selection	`opExcise, opMask, opRestriction, opSubsref`
Meta operators	`opBlockDiag, opConj, opDiag, opCTranspose, opKron opDictionary, opZeros, opImage, opInverse, opReal opMinus, opOrthogonal, opPInverse, opPower, opFoG opStack, opSubsAsgn, opSum, opTranspose opEmpty, opZeros`
Wrapping	`opClass, opFunction, opMatrix`

Spot Operator Library

Operator type	Matlab function
Fast operators	`opConvolve, opCurvelet, opDCT, opDCT2 opDFT, opDFT2, opDirac, opEye, opHaar, opHaar2, opHadamard, opHeaviside, opOnes, opToeplitz opToepGauss, opToepSign, opWavelet`
Ensembles	`opBernoulli, opBinary, opGaussian, opSparseBinary`
Selection	`opExcise, opMask, opRestriction, opSubsref`
Meta operators	`opBlockDiag, opConj, opDiag, opCTranspose, opKron opDictionary, opZeros, opImage, opInverse, opReal opMinus, opOrthogonal, opPInverse, opPower, opFoG opStack, opSubsAsgn, opSum, opTranspose opEmpty, opZeros`
Wrapping	`opClass, opFunction, opMatrix`

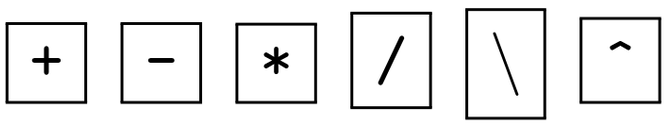
Spot Operator Library

Operator type	Matlab function
Fast operators	`opConvolve, opCurvelet, opDCT, opDCT2 opDFT, opDFT2, opDirac, opEye, opHaar, opHaar2, opHadamard, opHeaviside, opOnes, opToeplitz opToepGauss, opToepSign, opWavelet`
Ensembles	`opBernoulli, opBinary, opGaussian, opSparseBinary`
Selection	`opExcise, opMask, opRestriction, opSubsref`
Meta operators	`opBlockDiag, opConj, opDiag, opCTranspose, opKron opDictionary, opZeros, opImage, opInverse, opReal opMinus, opOrthogonal, opPInverse, opPower, opFoG opStack, opSubsAsgn, opSum, opTranspose opEmpty, opZeros`
Wrapping	`opClass, opFunction, opMatrix`



Spot Operator Library

Operator type	Matlab function
Fast operators	`opConvolve, opCurvelet, opDCT, opDCT2 opDFT, opDFT2, opDirac, opEye, opHaar, opHaar2, opHadamard, opHeaviside, opOnes, opToeplitz opToepGauss, opToepSign, opWavelet`
Ensembles	`opBernoulli, opBinary, opGaussian, opSparseBinary`
Selection	`opExcise, opMask, opRestriction, opSubsref`
Meta operators	`opBlockDiag, opConj, opDiag, opCTranspose, opKron opDictionary, opZeros, opImage, opInverse, opReal opMinus, opOrthogonal, opPInverse, opPower, opFoG opStack, opSubsAsgn, opSum, opTranspose opEmpty, opZeros`
Wrapping	`opClass, opFunction, opMatrix`

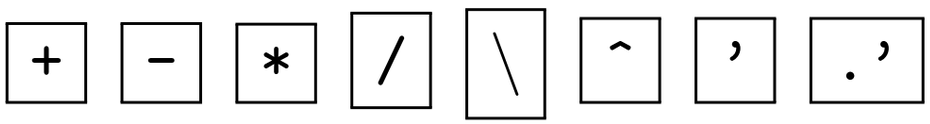


Spot Operator Library

Operator type	Matlab function
Fast operators	`opConvolve, opCurvelet, opDCT, opDCT2 opDFT, opDFT2, opDirac, opEye, opHaar, opHaar2, opHadamard, opHeaviside, opOnes, opToeplitz opToepGauss, opToepSign, opWavelet`
Ensembles	`opBernoulli, opBinary, opGaussian, opSparseBinary`
Selection	`opExcise, opMask, opRestriction, opSubsref`
Meta operators	`opBlockDiag, opConj, opDiag, opCTranspose, opKron opDictionary, opZeros, opImage, opInverse, opReal opMinus, opOrthogonal, opPInverse, opPower, opFoG opStack, opSubsAsgn, opSum, opTranspose opEmpty, opZeros`
Wrapping	`opClass, opFunction, opMatrix`

Spot Operator Library

Operator type	Matlab function
Fast operators	`opConvolve, opCurvelet, opDCT, opDCT2 opDFT, opDFT2, opDirac, opEye, opHaar, opHaar2, opHadamard, opHeaviside, opOnes, opToeplitz opToepGauss, opToepSign, opWavelet`
Ensembles	`opBernoulli, opBinary, opGaussian, opSparseBinary`
Selection	`opExcise, opMask, opRestriction, opSubsref`
Meta operators	`opBlockDiag, opConj, opDiag, opCTranspose, opKron opDictionary, opZeros, opImage, opInverse, opReal opMinus, opOrthogonal, opPInverse, opPower, opFoG opStack, opSubsAsgn, opSum, opTranspose opEmpty, opZeros`
Wrapping	`opClass, opFunction, opMatrix`



Spot Operator Library

Operator type	Matlab function
Fast operators	`opConvolve, opCurvelet, opDCT, opDCT2 opDFT, opDFT2, opDirac, opEye, opHaar, opHaar2, opHadamard, opHeaviside, opOnes, opToeplitz opToepGauss, opToepSign, opWavelet`
Ensembles	`opBernoulli, opBinary, opGaussian, opSparseBinary`
Selection	`opExcise, opMask, opRestriction, opSubsref`
Meta operators	`opBlockDiag, opConj, opDiag, opCTranspose, opKron opDictionary, opZeros, opImage, opInverse, opReal opMinus, opOrthogonal, opPInverse, opPower, opFoG opStack, opSubsAsgn, opSum, opTranspose opEmpty, opZeros`
Wrapping	`opClass, opFunction, opMatrix`

`+` `-` `*` `/` `\` `^` `'` `.'` `[.,.]`

Spot Operator Library

Operator type	Matlab function
Fast operators	`opConvolve, opCurvelet, opDCT, opDCT2 opDFT, opDFT2, opDirac, opEye, opHaar, opHaar2, opHadamard, opHeaviside, opOnes, opToeplitz opToepGauss, opToepSign, opWavelet`
Ensembles	`opBernoulli, opBinary, opGaussian, opSparseBinary`
Selection	`opExcise, opMask, opRestriction, opSubsref`
Meta operators	`opBlockDiag, opConj, opDiag, opCTranspose, opKron opDictionary, opZeros, opImage, opInverse, opReal opMinus, opOrthogonal, opPInverse, opPower, opFoG opStack, opSubsAsgn, opSum, opTranspose opEmpty, opZeros`
Wrapping	`opClass, opFunction, opMatrix`

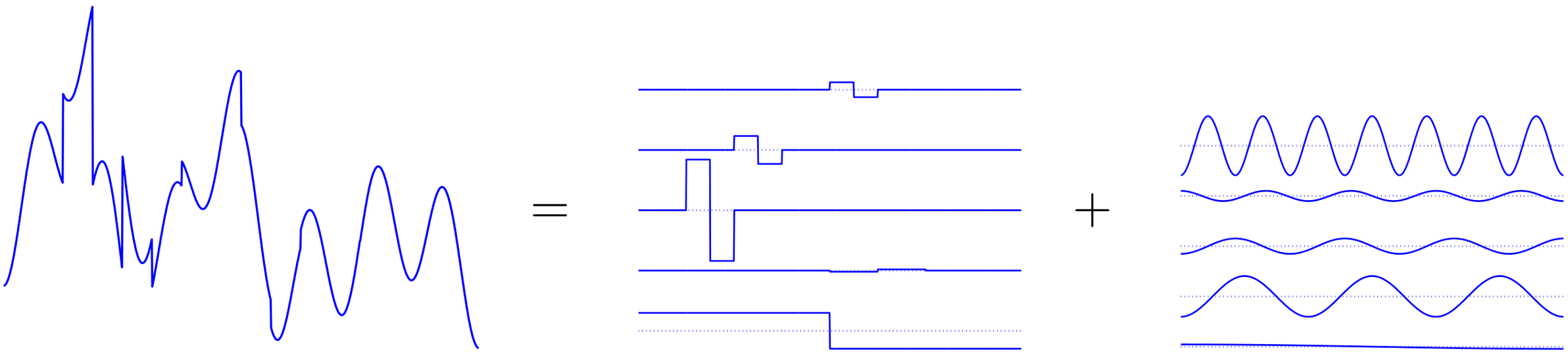
`+` `-` `*` `/` `\` `^` `'` `.'` `[.,.]` `[.;.]`

Spot Operator Library

Operator type	Matlab function
Fast operators	`opConvolve, opCurvelet, opDCT, opDCT2 opDFT, opDFT2, opDirac, opEye, opHaar, opHaar2, opHadamard, opHeaviside, opOnes, opToeplitz opToepGauss, opToepSign, opWavelet`
Ensembles	`opBernoulli, opBinary, opGaussian, opSparseBinary`
Selection	`opExcise, opMask, opRestriction, opSubsref`
Meta operators	`opBlockDiag, opConj, opDiag, opCTranspose, opKron opDictionary, opZeros, opImage, opInverse, opReal opMinus, opOrthogonal, opPInverse, opPower, opFoG opStack, opSubsAsgn, opSum, opTranspose opEmpty, opZeros`
Wrapping	`opClass, opFunction, opMatrix`

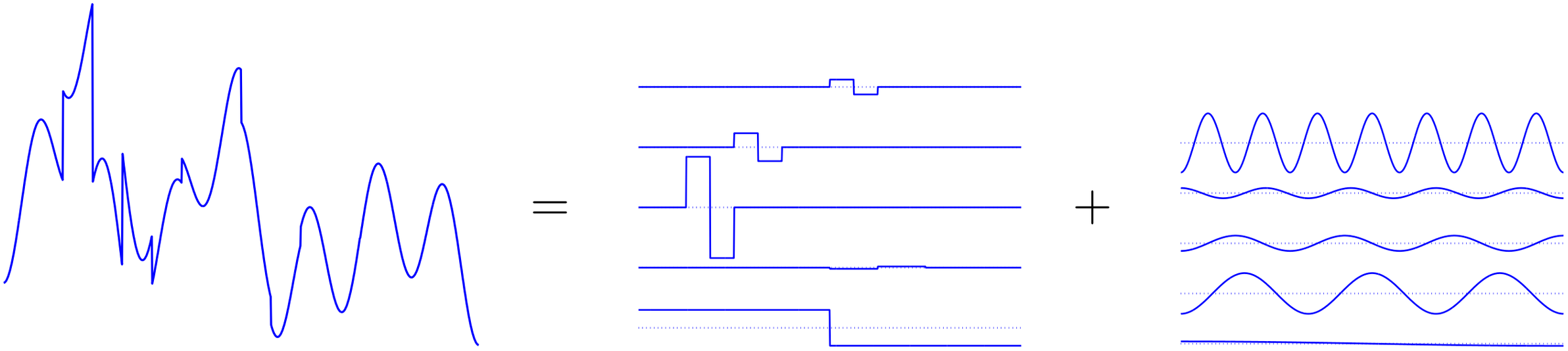
`+` `-` `*` `/` `\` `^` `'` `.'` `[.,.]` `[.;.]` `bicg char disp eigs isreal minres qmr subsasgn bicgstab conj display end imag isscalar pcg realsubsref blkdiag double full inv kron ndims pinv rrandn symmlq cgs diag drandn gmres isempty lsqr normest size`

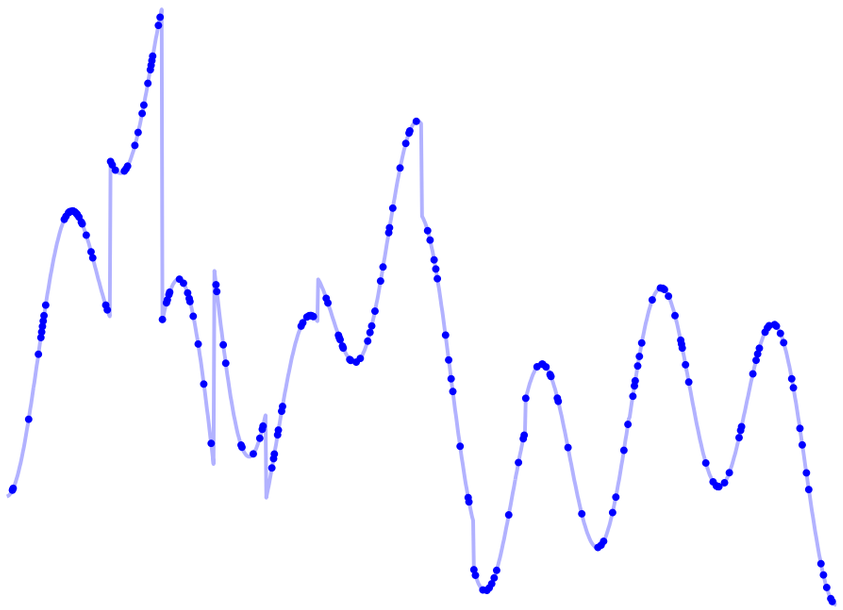
Example 2: Sparse Recovery



$$b \approx \boxed{\text{Haar}\ \ \text{DCT}}\ x$$

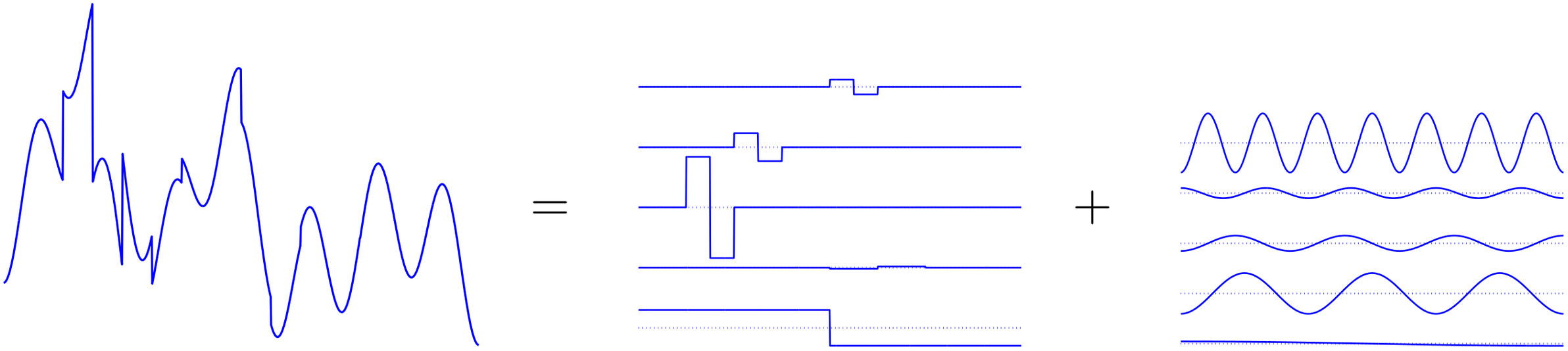
Example 2: Sparse Recovery



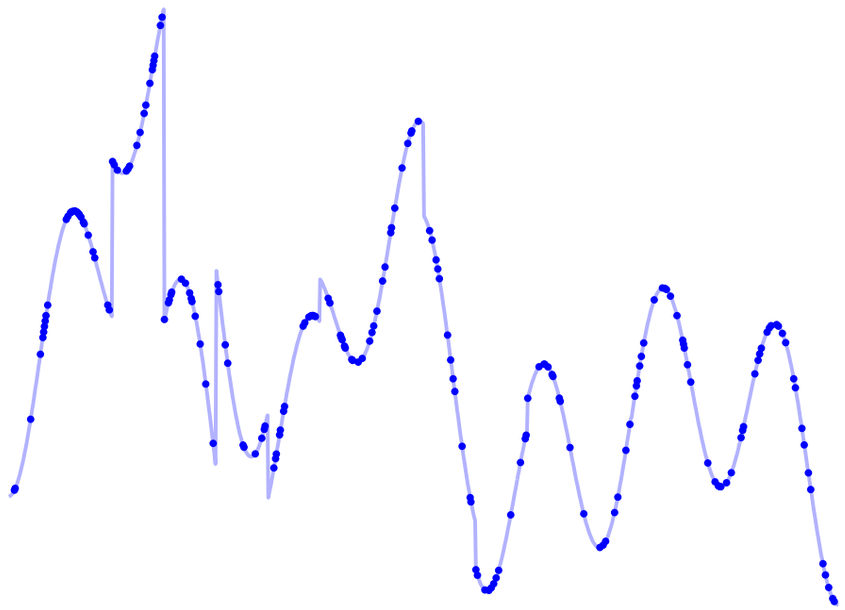


$$b \approx \boxed{\text{Haar}\,|\,\text{DCT}}\, x$$

Example 2: Sparse Recovery



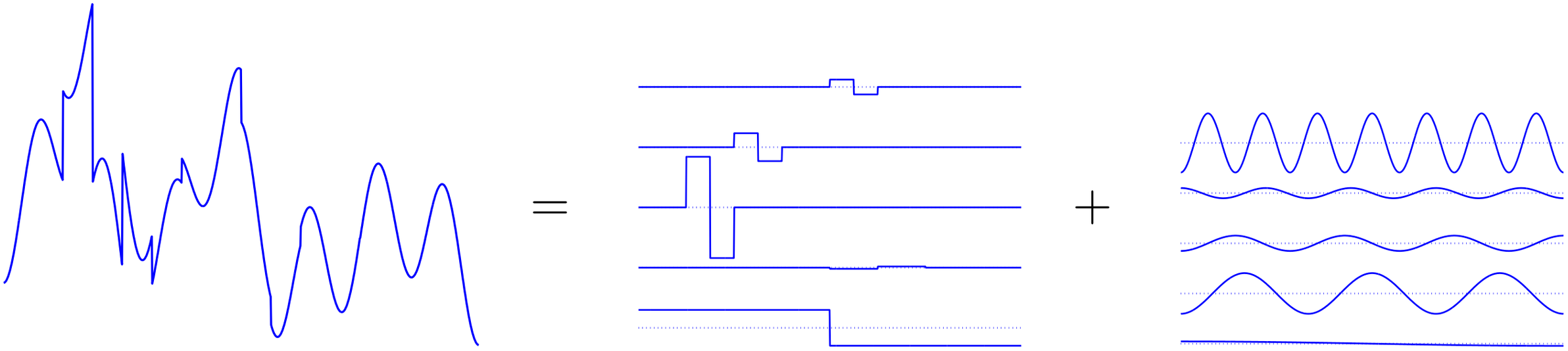
$\mathbf{R}b =$



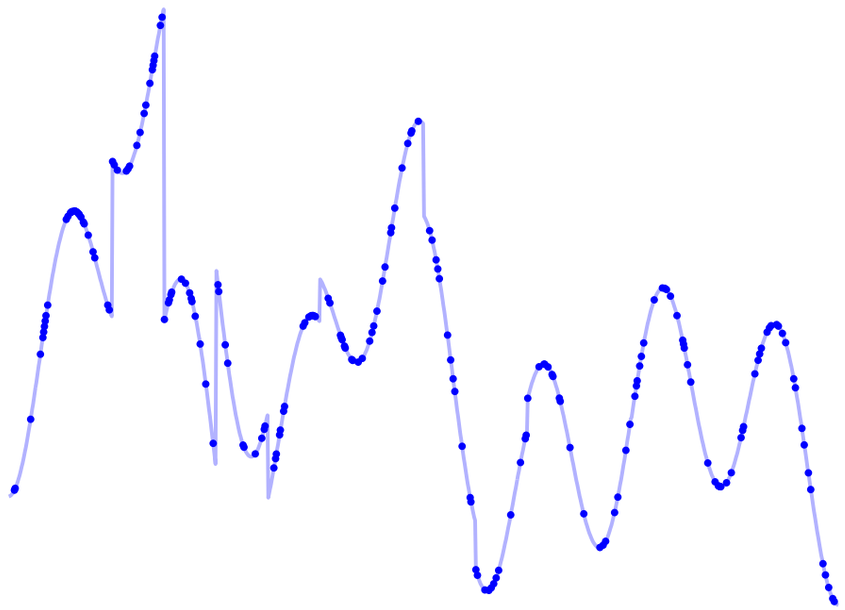
$\mathbf{R} =$ Restriction

$b \approx$ Haar | DCT x

Example 2: Sparse Recovery



$\mathbf{R}b =$



$\mathbf{R} =$ Restriction

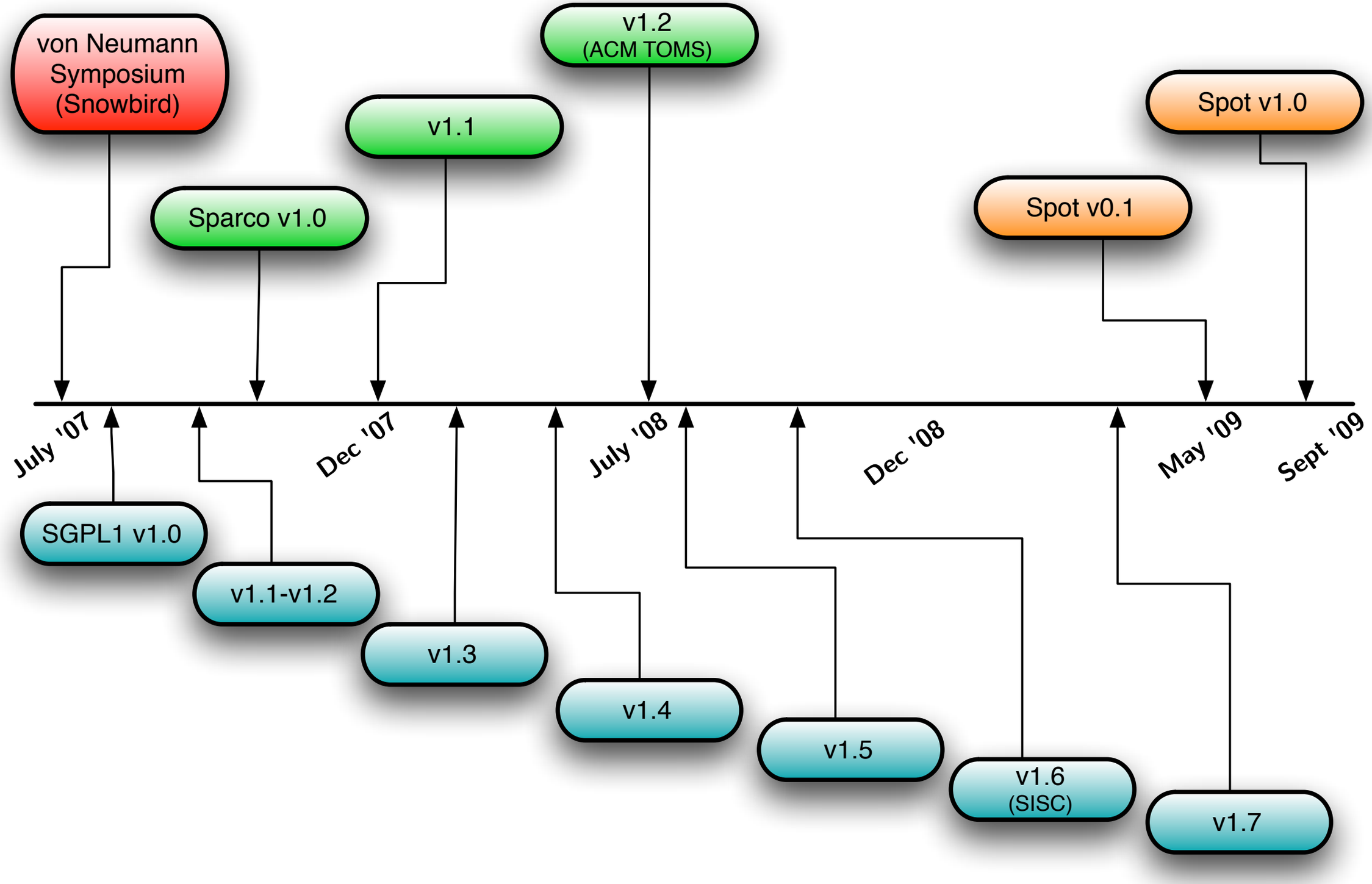
$\mathbf{R}b \approx \mathbf{R}$ Haar DCT x

Sparse Recovery Demo

Extensibility

```
classdef opDCT < opSpot
%  D = opDCT(N) creates a 1-D DCT for length-N signals
   methods
      function op = opDCT(n)
      %opDCT constructor.
         op = op@opSpot('DCT',n,n);
      end
      function y = multiply(op,x,mode)
      %multiply  apply the DCT operator.
         if mode == 1
            y = dct(x);    % ie, y = D*x
         else
            y = idct(x);   % ie, y = D'*x
         end
      end
   end % methods
end % classdef
```

Development History



Looking Ahead

Large-scale computing (Joint with Felix Herrmann, EOS)

- out-of-core operations — borrow ideas from SlimPy
- integrate with Parallel Computing Toolbox — MSc project

Looking Ahead

Large-scale computing (Joint with Felix Herrmann, EOS)

- out-of-core operations — borrow ideas from SlimPy
- integrate with Parallel Computing Toolbox — MSc project

Related projects

- bbtools (E. Høgh-Rasmussen) — last update: Apr 2006
- RestoreTools (J. Nagy) — last update: Mar 2007

Looking Ahead

Large-scale computing (Joint with Felix Herrmann, EOS)

- out-of-core operations — borrow ideas from SlimPy
- integrate with Parallel Computing Toolbox — MSc project

Related projects

- bbtools (E. Høgh-Rasmussen) — last update: Apr 2006
- RestoreTools (J. Nagy) — last update: Mar 2007

Code is available!

`http://www.cs.ubc.ca/labs/scl/spot`