Douglas–Rachford Feasibility Methods for Matrix Compl. . .

Laureate Prof. Jonathan Borwein with Matthew Tam
http://carma.newcastle.edu.au/DRmethods/paseky.html

THE UNIVERSITY OF NEWCASTLE AUSTRALIA

CARMA

Spring School on Variational Analysis VI
Paseky nad Jizerou, April 19–25, 2015

Last Revised: May 6, 2016

Those Involved

Note the fluid flow being studied

Fran Aragón   Jon Borwein

Matt Tam

Matrix Completion Preliminaries

Many successful non-convex applications of the Douglas–Rachford method can be considered as matrix completion problems (a well studied topic).

In the remainder of this series, we shall focus on recent successful applications of the method to a variety of (real) matrix reconstruction problems.

In particular, consider matrix completion in the context of:

1. Positive semi-definite matrices.
2. Stochastic matrices.
3. Euclidean distance matrices, esp. those in protein reconstruction.
4. Hadamard matrices together with their specialisations.
5. Nonograms – a Japanese number "painting" game.
6. Sudoku – a Japanese number game.

The framework is flexible and there are many other actual and potential applications. Our exposition will highlight the importance of the model.

Matrix Completion Preliminaries

Many successful non-convex applications of the Douglas–Rachford method can be considered as matrix completion problems (a well studied topic).

In the remainder of this series, we shall focus on recent successful applications of the method to a variety of (real) matrix reconstruction problems.

In particular, consider matrix completion in the context of:

1. Positive semi-definite matrices.
2. Stochastic matrices.
3. Euclidean distance matrices, esp. those in protein reconstruction.
4. Hadamard matrices together with their specialisations.
5. Nonograms – a Japanese number "painting" game.
6. Sudoku – a Japanese number game.

The framework is flexible and there are many other actual and potential applications. Our exposition will highlight the importance of the model.

Matrix Completion

From herein, we consider $\mathcal{H} = \mathbb{R}^{m\times n}$ equipped with the trace inner product and induced (Frobenius) norm:

$$\langle A, B\rangle := \operatorname{tr}(A^T B), \quad \|A\|_F := \sqrt{\operatorname{tr}(A^T A)} = \sqrt{\sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij}^2}.$$

- A partial matrix is an $m \times n$ array for which only entries in certain locations are known.
- A completion of the partial matrix $A = (a_{ij}) \in \mathbb{R}^{m\times n}$, is a matrix $B = (b_{ij}) \in \mathbb{R}^{m\times n}$ such that if a_{ij} is specified then $b_{ij} = a_{ij}$.

Abstractly matrix completion is the following:

> Given a partial matrix, find a completion which belongs to some prescribed family of matrices.

Matrix Completion

From herein, we consider $\mathcal{H} = \mathbb{R}^{m\times n}$ equipped with the trace inner product and induced (Frobenius) norm:

$$\langle A, B\rangle := \mathrm{tr}(A^T B), \quad \|A\|_F := \sqrt{\mathrm{tr}(A^T A)} = \sqrt{\sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij}^2}.$$

- A partial matrix is an $m \times n$ array for which only entries in certain locations are known.
- A completion of the partial matrix $A = (a_{ij}) \in \mathbb{R}^{m\times n}$, is a matrix $B = (b_{ij}) \in \mathbb{R}^{m\times n}$ such that if a_{ij} is specified then $b_{ij} = a_{ij}$.

Abstractly matrix completion is the following:

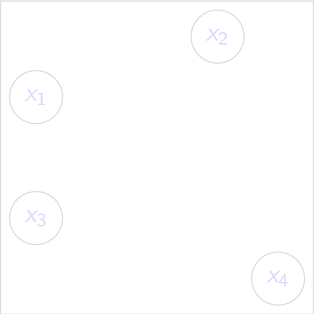
Given a partial matrix, find a completion which belongs to some prescribed family of matrices.

Matrix Completion: Example

Suppose the partial matrix $D = (D_{ij}) \in \mathbb{R}^{4\times 4}$ is known to contains the pair-wise distances between four points $x_1, \ldots, x_4 \in \mathbb{R}^2$. That is,

$$D_{ij} = \|x_i - x_j\|^2.$$

$$D = \begin{pmatrix} 0 & 3.1 & \mathbf{?} & \mathbf{?} \\ 3.1 & 0 & \mathbf{?} & \mathbf{?} \\ ? & ? & 0 & 4.3 \\ ? & ? & 4.3 & 0 \end{pmatrix}$$



four points in $\mathbb{R}^2$

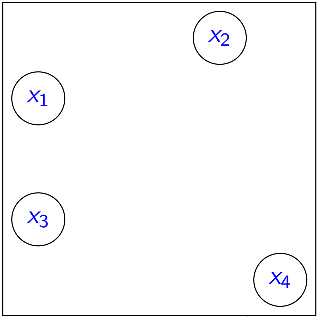
⟶ Reconstruct D from known entries and *a priori* information.

Matrix Completion: Example

Suppose the partial matrix $D = (D_{ij}) \in \mathbb{R}^{4\times 4}$ is known to contains the pair-wise distances between four points $x_1, \dots, x_4 \in \mathbb{R}^2$. That is,

$$D_{ij} = \|x_i - x_j\|^2.$$

$$D = \begin{pmatrix} 0 & 3.1 & \mathbf{?} & \mathbf{?} \\ 3.1 & 0 & \mathbf{?} & \mathbf{?} \\ ? & ? & 0 & 4.3 \\ ? & ? & 4.3 & 0 \end{pmatrix}$$



four points in $\mathbb{R}^2$

⟶ Reconstruct D from known entries and *a priori* information.

Matrix Completion: Example

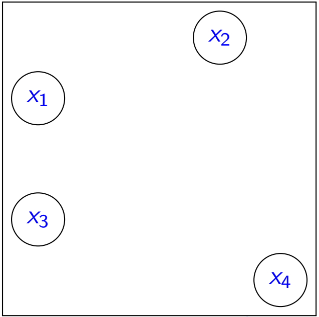
Suppose the partial matrix $D = (D_{ij}) \in \mathbb{R}^{4\times4}$ is known to contains the pair-wise distances between four points $x_1, \ldots, x_4 \in \mathbb{R}^2$. That is,

$$D_{ij} = \|x_i - x_j\|^2.$$

$$D = \begin{pmatrix} 0 & 3.1 & \mathbf{?} & \mathbf{?} \\ 3.1 & 0 & \mathbf{?} & \mathbf{?} \\ \mathbf{?} & \mathbf{?} & 0 & 4.3 \\ \mathbf{?} & \mathbf{?} & 4.3 & 0 \end{pmatrix}$$

$$??? \Downarrow$$

$$D = \begin{pmatrix} 0 & 3.1 & \mathbf{2.0} & \mathbf{5} \\ 3.1 & 0 & \mathbf{4.2} & \mathbf{4.1} \\ \mathbf{2.0} & \mathbf{4.2} & 0 & 4.3 \\ \mathbf{5} & \mathbf{4.1} & 4.3 & 0 \end{pmatrix}$$



four points in $\mathbb{R}^2$

$\longrightarrow$ Reconstruct D from known entries and *a priori* information.

Matrix Completion Preliminaries

It is natural to formulate matrix completions as the feasibility problem:

$$\text{find } X \in \bigcap_{i=1}^{N} C_i \subseteq \mathbb{R}^{m\times n}.$$

Let A be the partial matrix to be completed. We (mostly) choose

- C_1 to be the set of all matrix completions of A.
- $C_2, \dots, C_N$ s.t. their intersection equals the prescribed matrix family.

Let Ω denote the set of indices for the entry in A is known. Then

$$C_1 := \{X \in \mathbb{R}^{m\times n} : X_{ij} = A_{ij} \text{ for all } (i,j) \in \Omega\}.$$

The projection of $X \in \mathbb{R}^{m\times n}$ onto C_1 is given pointwise by

$$P_{C_1}(X)_{ij} = \begin{cases} A_{ij}, & \text{if } (i,j) \in \Omega, \\ X_{ij}, & \text{otherwise.} \end{cases}$$

The remainder of the talk will focus on choosing $C_2, \dots, C_N$.

Positive Semi-Definite Matrices

Denote the symmetric matrices by $\mathbb{S}^n$, and the positive semi-definite matrices by $\mathbb{S}^n_+$. Our second constraint set is

$$C_2 := \mathbb{S}^n_+ = \{X \in \mathbb{R}^{n\times n} : X = X^T, y^T X y \geq 0 \text{ for all } y \in \mathbb{R}^n\}.$$

The matrix X is a PSD completion of A if and only if $X \in C_1 \cap C_2$.

Theorem (Higham 1986)

For any $X \in \mathbb{R}^{n\times n}$, define $Y = (X + X^T)/2$ and let $Y = UP$ be a polar decomposition of Y (*i.e.*, U unitary, $P \in \mathbb{S}^n_+$.). Then

$$P_{C_2}(X) = \frac{Y + P}{2}.$$

An important class of PSD matrices are the correlation matrices.

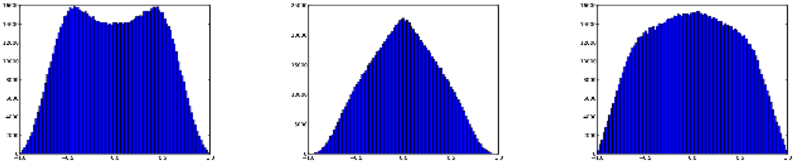
Positive Semi-Definite Matrices: Correlation Matrices

For random variables $X_1, X_2, \ldots, X_n$, the ij-th entry of the corresponding correlation matrix contains the correlation between X_i and X_j. This is incorporated into C_1 by enforcing that

$$(i,i) \in \Omega \text{ with } A_{ii} = 1 \text{ for } i = 1, 2, \ldots, n. \tag{1}$$

Moreover, whenever (1) holds for a matrix its entries are necessarily contained in $[-1, 1]$.

Apply this formulation for different starting points yields:



$X_0 := Y.$ $\quad X_0 := \frac{1}{2}(Y + Y^T) \in S_5.$ $\quad X_0 := YY^T \in S_5.$

Figure. Distribution of entries for correlation matrices generated by choosing different initial points. Y is a random matrix in $[-1, 1]^{5\times 5}$.

Stochastic matrices

Recall that a matrix $A = (A_{ij}) \in \mathbb{R}^{m \times n}$ is said to be doubly stochastic if

$$\sum_{i=1}^{m} A_{ij} = \sum_{j=1}^{n} A_{ij} = 1, A_{ij} \geq 0. \tag{2}$$

These matrices describe the transitions of a Markov chain (in this case $m = n$), amongst other things. We use the following constraint sets

$$C_2 := \left\{ X \in \mathbb{R}^{m \times n} | \sum_{i=1}^{m} X_{ij} = 1 \text{ for } j = 1, \ldots, n \right\},$$

$$C_3 := \left\{ X \in \mathbb{R}^{m \times n} | \sum_{j=1}^{n} X_{ij} = 1 \text{ for } i = 1, \ldots, m \right\},$$

$$C_4 := \{ X \in \mathbb{R}^{m \times n} | X_{ij} \geq 0 \text{ for } i = 1, \ldots, m \text{ and } j = 1, \ldots, n \}.$$

The matrix X is a double stochastic matrix completing A if and only if

$$X \in C_1 \cap C_2 \cap C_3 \cap C_4.$$

Stochastic matrices

Recall that a matrix $A = (A_{ij}) \in \mathbb{R}^{m \times n}$ is said to be doubly stochastic if

$$\sum_{i=1}^{m} A_{ij} = \sum_{j=1}^{n} A_{ij} = 1, A_{ij} \geq 0. \tag{2}$$

These matrices describe the transitions of a Markov chain (in this case $m = n$), amongst other things. We use the following constraint sets

$$C_2 := \left\{ X \in \mathbb{R}^{m \times n} | \sum_{i=1}^{m} X_{ij} = 1 \text{ for } j = 1, \ldots, n \right\},$$

$$C_3 := \left\{ X \in \mathbb{R}^{m \times n} | \sum_{j=1}^{n} X_{ij} = 1 \text{ for } i = 1, \ldots, m \right\},$$

$$C_4 := \{ X \in \mathbb{R}^{m \times n} | X_{ij} \geq 0 \text{ for } i = 1, \ldots, m \text{ and } j = 1, \ldots, n \}.$$

The matrix X is a double stochastic matrix completing A if and only if

$$X \in C_1 \cap C_2 \cap C_3 \cap C_4.$$

Stochastic matrices

$$C_2 := \left\{ X \in \mathbb{R}^{m\times n} \middle| \sum_{i=1}^{m} X_{ij} = 1 \text{ for } j = 1, \dots, n \right\},$$
$$C_4 := \{X \in \mathbb{R}^{m\times n} | X_{ij} \geq 0 \text{ for } i = 1, \dots, m \text{ and } j = 1, \dots, n\}.$$

Denote $\mathbf{e} = (1, 1, \dots, 1) \in \mathbb{R}^m$. Since C_2 applies to each column independently, a column-wise formula for P_{C_2} is given by

$$P_E(x) = x + \frac{1}{m}\left(1 - \sum_{i=1}^{m} x_j\right)\mathbf{e} \quad \text{where} \quad E := \{x \in \mathbb{R}^m : \mathbf{e}^T x = 1\}.$$

The projection of X onto C_4 is given pointwise by

$$P_{C_4}(X)_{ij} = \max\{0, X_{ij}\}.$$

- Singly stochastic matrix completion can be consider by dropping C_3.
- Related work of Thakouda applies Dykstra's algorithm to a two set model. The corresponding projections are less straight-forward.

Hadamard Matrices

A matrix $H = (H_{ij}) \in \{-1, 1\}^{n \times n}$ is said to be a Hadamard matrix of order n if [1]

$$H^T H = nI.$$

A classical result of Hadamard asserts that Hadamard matrices exist only if $n = 1, 2$ or a multiple of 4. For orders 1 and 2, such matrices are easy to find. For example,

$$[1], \quad \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}.$$

The (open) Hadamard conjecture is concerned with the converse:

> There exists a Hadamard matrices of order $4n$ for all $n \in \mathbb{N}$.

[1] There are many equivalent characterizations and many local experts.

Hadamard Matrices

A matrix $H = (H_{ij}) \in \{-1,1\}^{n\times n}$ is said to be a Hadamard matrix of order n if [1]

$$H^T H = nI.$$

A classical result of Hadamard asserts that Hadamard matrices exist only if $n = 1, 2$ or a multiple of 4. For orders 1 and 2, such matrices are easy to find. For example,

$$\begin{bmatrix}1\end{bmatrix}, \quad \begin{bmatrix}1 & -1\\ 1 & 1\end{bmatrix}.$$

The (open) Hadamard conjecture is concerned with the converse:

There exists a Hadamard matrices of order $4n$ for all $n \in \mathbb{N}$.

[1] There are many equivalent characterizations and many local experts.

Hadamard Matrices

Consider now the problem of finding a Hadamard matrix of a given order – an important completion problem with structure restriction but no fixed entries. We use the following constraint sets:

$$C_1 := \{X \in \mathbb{R}^{n\times n} | X_{ij} = \pm 1 \text{ for } i,j = 1,\dots,n\},$$
$$C_2 := \{X \in \mathbb{R}^{n\times n} | X^T X = nI\}.$$

Then X is a Hadamard matrix if and only if $X \in C_1 \cap C_2$.

The projection of X on C_1 is given by pointwise rounding to ± 1.

Proposition (A projection onto C_2)

Let $X = USV^T$ be a singular value decomposition. Then

$$\sqrt{n}UV^T \in P_{C_2}(X).$$

Hadamard Matrices

Consider now the problem of finding a Hadamard matrix of a given order – an important completion problem with structure restriction but no fixed entries. We use the following constraint sets:

$$C_1 := \{X \in \mathbb{R}^{n\times n} | X_{ij} = \pm 1 \text{ for } i, j = 1, \ldots, n\},$$
$$C_2 := \{X \in \mathbb{R}^{n\times n} | X^T X = nI\}.$$

Then X is a Hadamard matrix if and only if $X \in C_1 \cap C_2$.

The projection of X on C_1 is given by pointwise rounding to ± 1.

Proposition (A projection onto C_2)

Let $X = USV^T$ be a singular value decomposition. Then

$$\sqrt{n}UV^T \in P_{C_2}(X).$$

Hadamard Matrices

Let H_1 and H_2 be Hadamard matrices. We say H_1 are H_2 are:

- Distinct if $H_1 \neq H_2$,
- Equivalent if H_2 can be obtained from H_1 by performing row/column permutations, and/or multiplying rows/columns by -1.

For order $4n$:

- Number of Distinct Hadamard matrices is OEIS A206712:

$$768, 4954521600, 20251509535014912000, \ldots$$

- Number of Inequivalent Hadamard matrices is OEIS A00729:

$$1, 1, 1, 1, 5, 3, 60, 487, 13710027, \ldots$$

With increasing order, the number of Hadamard matrices is a faster than exponentially decreasing proportion of total number of ± 1–matrices (there are 2^{n^2} ± 1–matrices or order n).

Hadamard Matrices

Let H_1 and H_2 be Hadamard matrices. We say H_1 are H_2 are:

- Distinct if $H_1 \neq H_2$,
- Equivalent if H_2 can be obtained from H_1 by performing row/column permutations, and/or multiplying rows/columns by -1.

For order $4n$:

- Number of Distinct Hadamard matrices is OEIS A206712:

$$768, 4954521600, 20251509535014912000, \ldots$$

- Number of Inequivalent Hadamard matrices is OEIS A00729:

$$1, 1, 1, 1, 5, 3, 60, 487, 13710027, \ldots$$

With increasing order, the number of Hadamard matrices is a faster than exponentially decreasing proportion of total number of ± 1–matrices (there are 2^{n^2} ± 1–matrices or order n).

Hadamard Matrices

Let H_1 and H_2 be Hadamard matrices. We say H_1 are H_2 are:

- Distinct if $H_1 \neq H_2$,
- Equivalent if H_2 can be obtained from H_1 by performing row/column permutations, and/or multiplying rows/columns by -1.

For order $4n$:

- Number of Distinct Hadamard matrices is OEIS A206712:

$$768, 4954521600, 20251509535014912000, \dots$$

- Number of Inequivalent Hadamard matrices is OEIS A00729:

$$1, 1, 1, 1, 5, 3, 60, 487, 13710027, \dots$$

With increasing order, the number of Hadamard matrices is a faster than exponentially decreasing proportion of total number of ±1–matrices (there are 2^{n^2} ±1–matrices or order n).

Hadamard Matrices

Table: Number of Hadamard matrices found from 1000 instances

Order	$C_1 \cap C_2$ Formulation			
	Ave Time (s)	Solved	Distinct	Inequivalent
2	1.1371	534	8	1
4	1.0791	627	422	1
8	0.7368	996	996	1
12	7.1298	0	0	0
16	9.4228	0	0	0
20	20.6674	0	0	0

Checking if two Hadamard matrices are equivalent can be cast as a problem of graph isomorphism (McKay '79).

- In Sage use `is_isomorphic(graph1,graph2)`.

Hadamard Matrices

We give an alternative formulation. Define:

$$C_1 := \{X \in \mathbb{R}^{n\times n} | X_{ij} = \pm 1 \text{ for } i, j = 1, \dots, n\},$$
$$C_3 := \{X \in \mathbb{R}^{n\times n} | X^T X = \|X\|_F I\}.$$

Then X is a Hadamard matrix if and only if $X \in C_1 \cap C_2 = C_1 \cap C_3$.

Proposition (A projection onto C_3)

Let $X = USV^T$ be a singular value decomposition. Then

$$\sqrt{\|X\|_F}\,UV^T \in P_{C_3}(X).$$

Hadamard Matrices

Table: Number of Hadamard matrices found from 1000 instances

Order	$C_1 \cap C_2$ Formulation			
	Ave Time (s)	Solved	Distinct	Inequivalent
2	1.1371	534	8	1
4	1.0791	627	422	1
8	0.7368	996	996	1
12	7.1298	**0**	**0**	0
16	9.4228	0	0	0
20	20.6674	0	0	0

Order	$C_1 \cap C_3$ Formulation			
	Ave Time (s)	Solved	Distinct	Inequivalent
2	1.1970	505	8	1
4	0.2647	921	541	1
8	0.0117	1000	1000	1
12	0.8337	**1000**	**1000**	1
16	11.7096	16	16	4
20	22.6034	0	0	0

- A more obvious formulation is can be less effective.

Hadamard Matrices

Table: Number of Hadamard matrices found from 1000 instances

Order	$C_1 \cap C_2$ Formulation			
	Ave Time (s)	Solved	Distinct	Inequivalent
2	1.1371	534	8	1
4	1.0791	627	422	1
8	0.7368	996	996	1
12	7.1298	**0**	**0**	0
16	9.4228	0	0	0
20	20.6674	0	0	0

Order	$C_1 \cap C_3$ Formulation			
	Ave Time (s)	Solved	Distinct	Inequivalent
2	1.1970	505	8	1
4	0.2647	921	541	1
8	0.0117	1000	1000	1
12	0.8337	**1000**	**1000**	1
16	11.7096	16	16	4
20	22.6034	0	0	0

- A more obvious formulation is can be less effective.

Skew-Hadamard Matrices

Recall that a matrix $X \in \mathbb{R}^{n \times n}$ is skew-symmetric if $X^T = -X$. A skew-Hadamard matrix is a Hadamard matrix H such that $(I - H)$ is skew-symmetric. That is,

$$H + H^T = 2I.$$

Skew-Hadamard matrices are of interest, for example, in the construction of various combinatorial designs. The number of inequivalent skew-Hadamard matrices of order $4n$ is OEIS A001119 (for $n = 2, 3, \ldots$):

$$1, 1, 2, 2, 16, 54, \ldots$$

It is convenient to redefine the constraint C_1 to be

$$C_1 = \{X \in \mathbb{R}^{n \times n} | X + X^T = 2I, \, X_{ij} = \pm 1 \text{ for } i, j = 1, \ldots, n\}.$$

A projection of X onto C_1 is given pointwise by

$$P_{C_1}(X) = \begin{cases} -1 & \text{if } i \neq j \text{ and } X_{ij} < X_{ji}, \\ 1 & \text{otherwise.} \end{cases}$$

Skew-Hadamard Matrices

Table: Number of skew-Hadamard matrices found from 1000 instances

Order	$C_1 \cap C_2$ Formulation			
	Ave Time (s)	Solved	Distinct	Inequivalent
2	0.0003	1000	2	1
4	1.1095	719	16	1
8	0.7039	902	889	1
12	14.1835	43	43	1
16	19.3462	0	0	0
20	29.0383	0	0	0

Order	$C_1 \cap C_3$ Formulation			
	Ave Time (s)	Solved	Distinct	Inequivalent
2	0.0004	1000	2	1
4	1.6381	634	16	1
8	0.0991	986	968	1
12	0.0497	999	999	1
16	0.2298	1000	1000	2
20	20.0296	495	495	2

- Adding constraints can help.

Skew-Hadamard Matrices

Table: Number of skew-Hadamard matrices found from 1000 instances

Order	$C_1 \cap C_2$ Formulation			
	Ave Time (s)	Solved	Distinct	Inequivalent
2	0.0003	1000	2	1
4	1.1095	719	16	1
8	0.7039	902	889	1
12	14.1835	43	43	1
16	19.3462	0	0	0
20	29.0383	0	0	0

Order	$C_1 \cap C_3$ Formulation			
	Ave Time (s)	Solved	Distinct	Inequivalent
2	0.0004	1000	2	1
4	1.6381	634	16	1
8	0.0991	986	968	1
12	0.0497	999	999	1
16	0.2298	1000	1000	2
20	20.0296	495	495	2

- Adding constraints can help.

Sudoku Puzzles

In Sudoku the player fills entries of an incomplete Latin square subject to the constraints:

- Each row contains the numbers 1 through 9 exactly once.
- Each column contains the numbers 1 through 9 exactly once.
- Each 3×3 sub-block contains the numbers 1 through 9 exactly once.

		5	3					
8							2	
	7			1		5		
4					5	3		
	1			7				6
		3	2				8	
	6		5					9
		4					3	
					9	7		

1	4	5	3	2	7	6	9	8
8	3	9	6	5	4	1	2	7
6	7	2	9	1	8	5	4	3
4	9	6	1	8	5	3	7	2
2	1	8	4	7	3	9	5	6
7	5	3	2	9	6	4	8	1
3	6	7	5	4	2	1	8	9
9	8	4	7	6	1	2	3	5
5	2	1	8	3	9	7	6	4

Figure. An incomplete Sudoku (left) and its unique solution (right).

- The Douglas–Rachford algorithm applied to the natural integer feasibility problem fails (exception: $n^2 \times n^2$ Sudokus where $n = 1, 2$).

Sudoku Puzzles: A Binary Model[5]

Let $E = \{e_j\}_{j=1}^{9} \subset \mathbb{R}^9$ be the standard basis. Define $X \in \mathbb{R}^{9\times 9\times 9}$ by

$$X_{ijk} = \begin{cases} 1 & \text{if } ij\text{th entry of the Sudoku is } k, \\ 0 & \text{otherwise.} \end{cases}$$

The idea: Reformulate integer entries as binary vectors.

7					9		5	
	1						3	
		2	3			7		
		4	5				7	
8						2		
					6	4		
	9			1				
	8			6				
		5	4					7

The constraints are:

$C_1 = \{X : X_{ij} \in E\}$

$C_2 = \{X : X_{ik} \in E\}$

$C_3 = \{X : X_{jk} \in E\}$

$C_4 = \{X : \text{vec}(3 \times 3 \text{ submatrix}) \in E\}$

$C_5 = \{X : X \text{ matches original puzzle}\}$

A solution is any $X \in \bigcap_{i=1}^{5} C_i$.

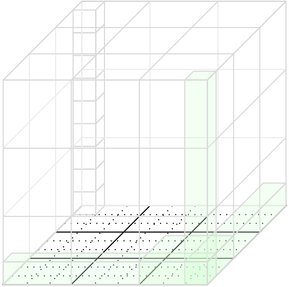
[5] Veit Elser was the first to realise the usefulness of this binary formulation for

Sudoku Puzzles: A Binary Model[5]

Let $E = \{e_j\}_{j=1}^{9} \subset \mathbb{R}^9$ be the standard basis. Define $X \in \mathbb{R}^{9\times 9\times 9}$ by

$$X_{ijk} = \begin{cases} 1 & \text{if } ij\text{th entry of the Sudoku is } k, \\ 0 & \text{otherwise.} \end{cases}$$

The idea: Reformulate integer entries as binary vectors.



The constraints are:

$C_1 = \{X : X_{ij} \in E\}$

$C_2 = \{X : X_{ik} \in E\}$

$C_3 = \{X : X_{jk} \in E\}$

$C_4 = \{X : \text{vec}(3 \times 3 \text{ submatrix}) \in E\}$

$C_5 = \{X : X \text{ matches original puzzle}\}$

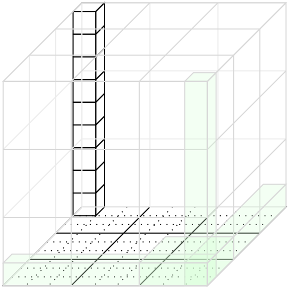
A solution is any $X \in \bigcap_{i=1}^{5} C_i$.

Sudoku Puzzles: A Binary Model[5]

Let $E = \{e_j\}_{j=1}^{9} \subset \mathbb{R}^9$ be the standard basis. Define $X \in \mathbb{R}^{9\times9\times9}$ by

$$X_{ijk} = \begin{cases} 1 & \text{if } ij\text{th entry of the Sudoku is } k, \\ 0 & \text{otherwise.} \end{cases}$$

The idea: Reformulate integer entries as binary vectors.



The constraints are:

$C_1 = \{X : X_{ij} \in E\}$

$C_2 = \{X : X_{ik} \in E\}$

$C_3 = \{X : X_{jk} \in E\}$

$C_4 = \{X : \text{vec}(3 \times 3 \text{ submatrix}) \in E\}$

$C_5 = \{X : X \text{ matches original puzzle}\}$

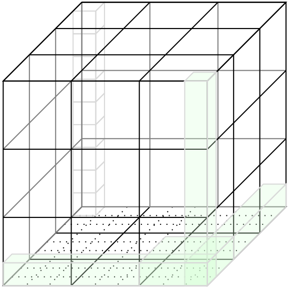
A solution is any $X \in \bigcap_{i=1}^{5} C_i$.

Sudoku Puzzles: A Binary Model[5]

Let $E = \{e_j\}_{j=1}^{9} \subset \mathbb{R}^9$ be the standard basis. Define $X \in \mathbb{R}^{9\times 9\times 9}$ by

$$X_{ijk} = \begin{cases} 1 & \text{if } ij\text{th entry of the Sudoku is } k, \\ 0 & \text{otherwise.} \end{cases}$$

The idea: Reformulate integer entries as binary vectors.



The constraints are:

$$C_1 = \{X : X_{ij} \in E\}$$
$$C_2 = \{X : X_{ik} \in E\}$$
$$C_3 = \{X : X_{jk} \in E\}$$
$$C_4 = \{X : \text{vec}(3 \times 3 \text{ submatrix}) \in E\}$$
$$C_5 = \{X : X \text{ matches original puzzle}\}$$

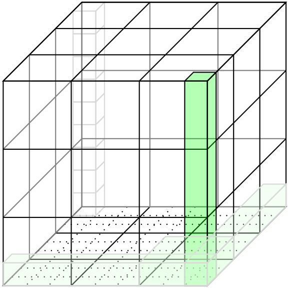
A solution is any $X \in \bigcap_{i=1}^{5} C_i$.

Sudoku Puzzles: A Binary Model[5]

Let $E = \{e_j\}_{j=1}^{9} \subset \mathbb{R}^9$ be the standard basis. Define $X \in \mathbb{R}^{9\times 9\times 9}$ by

$$X_{ijk} = \begin{cases} 1 & \text{if } ij\text{th entry of the Sudoku is } k, \\ 0 & \text{otherwise.} \end{cases}$$

The idea: Reformulate integer entries as binary vectors.



The constraints are:

$C_1 = \{X : X_{ij} \in E\}$

$C_2 = \{X : X_{ik} \in E\}$

$C_3 = \{X : X_{jk} \in E\}$

$C_4 = \{X : \text{vec}(3 \times 3 \text{ submatrix}) \in E\}$

$C_5 = \{X : X \text{ matches original puzzle}\}$

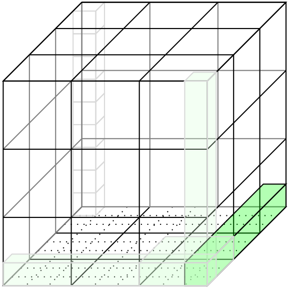
A solution is any $X \in \bigcap_{i=1}^{5} C_i$.

Sudoku Puzzles: A Binary Model[5]

Let $E = \{e_j\}_{j=1}^{9} \subset \mathbb{R}^9$ be the standard basis. Define $X \in \mathbb{R}^{9\times9\times9}$ by

$$X_{ijk} = \begin{cases} 1 & \text{if } ij\text{th entry of the Sudoku is } k, \\ 0 & \text{otherwise.} \end{cases}$$

The idea: Reformulate integer entries as binary vectors.



The constraints are:

$C_1 = \{X : X_{ij} \in E\}$

$C_2 = \{X : X_{ik} \in E\}$

$C_3 = \{X : X_{jk} \in E\}$

$C_4 = \{X : \mathrm{vec}(3 \times 3 \text{ submatrix}) \in E\}$

$C_5 = \{X : X \text{ matches original puzzle}\}$

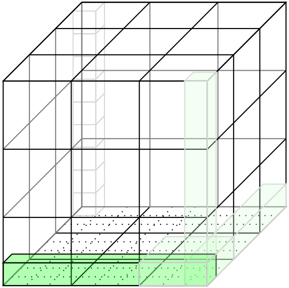
A solution is any $X \in \bigcap_{i=1}^{5} C_i$.

Sudoku Puzzles: A Binary Model[5]

Let $E = \{e_j\}_{j=1}^{9} \subset \mathbb{R}^9$ be the standard basis. Define $X \in \mathbb{R}^{9\times 9\times 9}$ by

$$X_{ijk} = \begin{cases} 1 & \text{if } ij\text{th entry of the Sudoku is } k, \\ 0 & \text{otherwise.} \end{cases}$$

The idea: Reformulate integer entries as binary vectors.



The constraints are:

$C_1 = \{X : X_{ij} \in E\}$

$C_2 = \{X : X_{ik} \in E\}$

$C_3 = \{X : X_{jk} \in E\}$

$C_4 = \{X : \text{vec}(3 \times 3 \text{ submatrix}) \in E\}$

$C_5 = \{X : X \text{ matches original puzzle}\}$

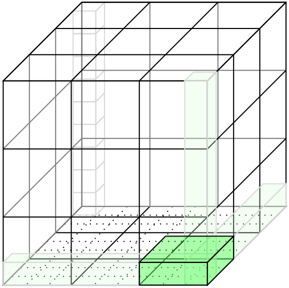
A solution is any $X \in \bigcap_{i=1}^{5} C_i$.

Sudoku Puzzles: A Binary Model[5]

Let $E = \{e_j\}_{j=1}^{9} \subset \mathbb{R}^9$ be the standard basis. Define $X \in \mathbb{R}^{9\times 9\times 9}$ by

$$X_{ijk} = \begin{cases} 1 & \text{if } ij\text{th entry of the Sudoku is } k, \\ 0 & \text{otherwise.} \end{cases}$$

The idea: Reformulate integer entries as binary vectors.



The constraints are:

$C_1 = \{X : X_{ij} \in E\}$

$C_2 = \{X : X_{ik} \in E\}$

$C_3 = \{X : X_{jk} \in E\}$

$C_4 = \{X : \text{vec}(3 \times 3 \text{ submatrix}) \in E\}$

$C_5 = \{X : X \text{ matches original puzzle}\}$

A solution is any $X \in \bigcap_{i=1}^{5} C_i$.

Sudoku Puzzles: Computing projections

Proposition (projections onto permutation sets)

Denote by $\mathcal{C} \subset \mathbb{R}^m$ the set of all vector whose entries are permutations of $c_1, c_2, \ldots, c_m \in \mathbb{R}$. Then for any $x \in \mathbb{R}^m$,

$$P_{\mathcal{C}} x = [\mathcal{C}]_x,$$

where $[\mathcal{C}]_x$ is the set of vectors $y \in \mathcal{C}$ such that ith largest index of y has the same index in y as the ith largest entry of x, for all indices i.

- $[\mathcal{C}]_x$ be computed efficiently using sorting algorithms.
- Choosing $c_1 = 1$ and $c_2 = \cdots = c_m = 0$ gives[2]

$$P_E x = \{e_i : x_i = \max\{x_1, \ldots, x_m\}\}.$$

Formulae for $P_{C_1}, P_{C_2}, P_{C_3}$ and P_{C_4} easily follow.

- P_{C_5} is given by setting the entries corresponding to those in the incomplete puzzle to 1, and leaving the remaining untouched.

[2] A direct proof of this special case appears in Jason Schaad's Masters thesis.

Sudoku Puzzles:Algorithm Details

1. Initialize: $x_0 := (y, y, y, y, y) \in D$ for some random $y \in [0, 1]^{9\times9\times9}$.
2. Iteration: By setting

$$x_{n+1} := T_{D,C}x_n = \frac{x_n + R_C R_D x_n}{2}.$$

3. Termination: Either if a solution is found, or 10000 iteration have been performed. More precisely, $\text{round}(P_D x_n)$ ($P_D x_n$ pointwise rounded to the nearest integer) is a solution if

$$\text{round}(P_D x_n) \in C \cap D.$$

Taking $\text{round}(\cdot)$ is valid since the solution is binary.

Sudoku Puzzles: An Experiment

We consider the following test libraries frequently used by programmers to test their solvers.

1. Dukuso's top95 and top1465.
2. First 1000 puzzles from Gordan Royle's minimum Sudoku – puzzles with 17 entries (the best known lower bound on the entries required for a unique solution).
3. reglib-1.3 – 1000 test puzzle suited to particular human style techniques.
4. ksudoku16 and ksudoku25 – a collection around 30 instances (various difficulties) generated with *KSudoku*. Contains larger 16×16 and 25×25 puzzles.[3]

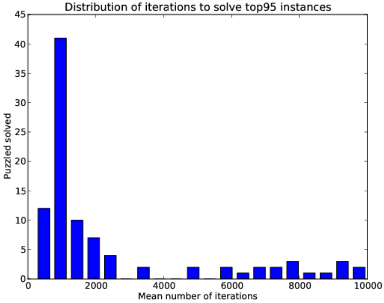
[3] Generating "hard" instances is a difficult problem.

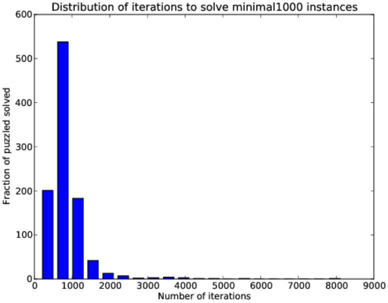
Computational Results: Success Rate

From 10 random replications of each puzzle:

Table. % Solved by the Douglas–Rachford method

top95	top1465	reglib-1.3	minimal1000	ksudoku16	ksudoku25
86.53	93.69	99.35	99.59	92	100





- If a instance was solved, the solution was usually found within the first 2000 iterations.

Computational Example: A 'Nasty' Sudoku

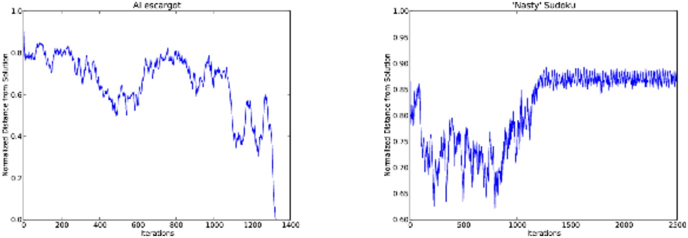
This 'nasty' Sudoku[4] cannot be solved reliably (20.2% success rate) by the Douglas–Rachford method.

7					9		5	
	1						3	
		2	3			7		
		4	5				7	
8						2		
					6	4		
	9			1				
	8			6				
		5	4					7

Other "difficult" Sudoku puzzles do not cause the Douglas–Rachford method any trouble.

- AI escargot = 98.5% success rate.

Figure. Distance to the solution by iterations



[4] This is a modified version of an example due to Veit Elser.

Computational Example: A ‘Nasty’ Sudoku

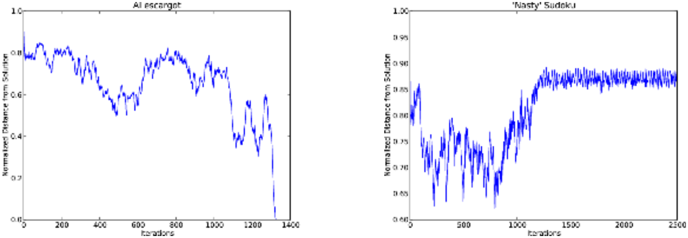
This ‘nasty’ Sudoku[4] cannot be solved reliably (20.2% success rate) by the Douglas–Rachford method.

7					9		5	
	1						3	
		2	3			7		
		4	5				7	
8						2		
					6	4		
	9			1				
	8			6				
		5	4					7

Other “difficult” Sudoku puzzles do not cause the Douglas–Rachford method any trouble.

- AI escargot = 98.5% success rate.

Figure. Distance to the solution by iterations



[4] This is a modified version of an example due to Veit Elser.

Computational Example: A 'Nasty' Sudoku

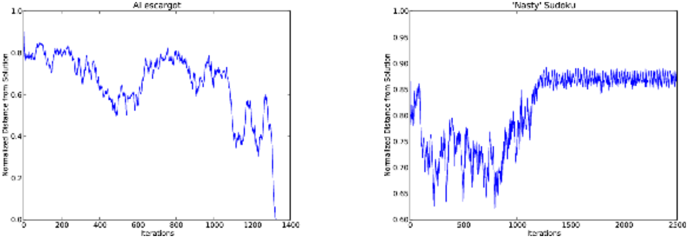
This 'nasty' Sudoku[4] cannot be solved reliably (20.2% success rate) by the Douglas–Rachford method.

7					9		5	
	1						3	
		2	3			7		
		4	5				7	
8						2		
					6	4		
	9			1				
	8			6				
		5	4					7

Other "difficult" Sudoku puzzles do not cause the Douglas–Rachford method any trouble.

- AI escargot = 98.5% success rate.

Figure. Distance to the solution by iterations



[4] This is a modified version of an example due to Veit Elser.

Computational Example: A ‘Nasty’ Sudoku

We considered solving the puzzles obtained by removing any single entry from the ‘Nasty’ Sudoku.

7					9		5	
	1						3	
		2	3			7		
		4	5				7	
8						2		
					6	4		
	9			1				
	8			6				
		5	4					7

Success rate when any single entry is removed:

- Top left 7 = 24%
- Any other entry = 99%

Number of solutions when any single entry is removed:

- Top left 7 = 5
- Any other entry = 200–3800

Is the Douglas–Rachford method hindered by an abundance of ‘near’ solutions?

Computational Example: A ‘Nasty’ Sudoku

We considered solving the puzzles obtained by removing any single entry from the ‘Nasty’ Sudoku.

7					9		5	
	1						3	
		2	3			7		
		4	5				7	
8						2		
					6	4		
	9			1				
	8			6				
		5	4					7

Success rate when any single entry is removed:

- Top left 7 = 24%
- Any other entry = 99%

Number of solutions when any single entry is removed:

- Top left 7 = 5
- Any other entry = 200–3800

Is the Douglas–Rachford method hindered by an abundance of ‘near’ solutions?

Computational Example: A ‘Nasty’ Sudoku

We considered solving the puzzles obtained by removing any single entry from the ‘Nasty’ Sudoku.

7					9		5	
	1						3	
		2	3			7		
		4	5				7	
8						2		
					6	4		
	9			1				
	8			6				
		5	4					7

Success rate when any single entry is removed:

- Top left 7 = 24%
- Any other entry = 99%

Number of solutions when any single entry is removed:

- Top left 7 = 5
- Any other entry = 200–3800

Is the Douglas–Rachford method hindered by
an abundance of ‘near’ solutions?

Computational Results: Performance Comparison

We compared the Douglas–Rachford method to the following solvers:

1. Gurobi binary program – Solves the same binary model using integer programming techniques.
2. YASS (Yet another Sudoku solver) – First applies a reasoning algorithm to determine possible candidates for each empty square. If this does not completely solve the puzzle, a deterministic recursive algorithm is used.
3. DLX – Solves an exact cover formulation using the *Dancing Links* implementation of Knuth's *Algorithm X* (non-deterministic, depth-first, back-tracking).

Table. Average Runtime (seconds).[5]

	top95	reglib-1.3	minimal1000	ksudoku16	ksudoku25
DR	1.432	0.279	0.509	5.064	4.011
Gurobi	0.063	0.059	0.063	0.168	0.401
YASS	2.256	0.039	0.654	-	-
DLX	1.386	0.105	3.871	-	-

[5] Some solvers are only designed to handle 9×9 puzzles.

Nonograms

A nonogram puzzle consists of a blank $m \times n$ grid of “pixels” together with $(m+n)$ cluster-size sequences (*i.e.*, for each row and each column). The goal is to “paint” the canvas with a picture such that:

1. Each pixel must be either black or white.
2. If a row (resp. column) has a cluster-size sequences $s_1, \ldots, s_k$ then it must contain k cluster of black pixels, each separated by at least one white pixel. The ith leftmost (resp. uppermost) cluster contains s_i black pixels.

						1			
			2			4	1	2	2
2	3	1	1	5	4	1	5	2	1

1	2										
	2										
	1										
	1										
	2										
2	4										
2	6										
	8										
1	1										
2	2										

Nonograms

A nonogram puzzle consists of a blank $m \times n$ grid of "pixels" together with $(m+n)$ cluster-size sequences (*i.e.*, for each row and each column). The goal is to "paint" the canvas with a picture such that:

1. Each pixel must be either black or white.
2. If a row (resp. column) has a cluster-size sequences $s_1, \dots, s_k$ then it must contain k cluster of black pixels, each separated by at least one white pixel. The ith leftmost (resp. uppermost) cluster contains s_i black pixels.

								1			
					2			4	1	2	2
		2	3	1	1	5	4	1	5	2	1
1	2										
	2										
	1										
	1										
	2										
2	4										
2	6										
	8										
1	1										
2	2										

Nonograms

A nonogram puzzle consists of a blank $m \times n$ grid of "pixels" together with $(m+n)$ cluster-size sequences (*i.e.*, for each row and each column). The goal is to "paint" the canvas with a picture such that:

1. Each pixel must be either black or white.
2. If a row (resp. column) has a cluster-size sequences $s_1, \ldots, s_k$ then it must contain k cluster of black pixels, each separated by at least one white pixel. The ith leftmost (resp. uppermost) cluster contains s_i black pixels.

								1			
					2			4	1	2	2
		2	3	1	1	5	4	1	5	2	1
1	2	■	□	■	■	□	□	□	□	□	□
	2										
	1										
	1										
	2										
2	4										
2	6										
	8										
1	1										
2	2										

Legal row

Nonograms

A nonogram puzzle consists of a blank $m \times n$ grid of “pixels” together with $(m+n)$ cluster-size sequences (*i.e.*, for each row and each column). The goal is to “paint” the canvas with a picture such that:

1. Each pixel must be either black or white.
2. If a row (resp. column) has a cluster-size sequences $s_1, \ldots, s_k$ then it must contain k cluster of black pixels, each separated by at least one white pixel. The ith leftmost (resp. uppermost) cluster contains s_i black pixels.

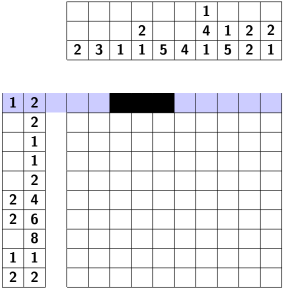
								1			
					2			4	1	2	2
		2	3	1	1	5	4	1	5	2	1
1	2				■			■	■		
	2										
	1										
	1										
	2										
2	4										
2	6										
	8										
1	1										
2	2										

Legal row

Nonograms

A nonogram puzzle consists of a blank $m \times n$ grid of "pixels" together with $(m+n)$ cluster-size sequences (*i.e.*, for each row and each column). The goal is to "paint" the canvas with a picture such that:

1. Each pixel must be either black or white.
2. If a row (resp. column) has a cluster-size sequences $s_1, \ldots, s_k$ then it must contain k cluster of black pixels, each separated by at least one white pixel. The ith leftmost (resp. uppermost) cluster contains s_i black pixels.



						1			
			2			4	1	2	2
2	3	1	1	5	4	1	5	2	1

1	2
	2
	1
	1
	2
2	4
2	6
	8
1	1
2	2

Illegal row

Nonograms

We model nonograms as a binary feasibility problem. The $m \times n$ grid is represented as a matrix $A \in \mathbb{R}^{m \times n}$ with

$$A[i,j] = \begin{cases} 0 & \text{if the } (i,j)\text{-th entry of the grid is white,} \\ 1 & \text{if the } (i,j)\text{-th entry of the grid is black.} \end{cases}$$

Let $\mathcal{R}_i \subset \mathbb{R}^m$ (resp. $\mathcal{C}_j \subset \mathbb{R}^n$) denote the set of vectors having cluster-size sequences matching row i (resp. column j). The constraints are:

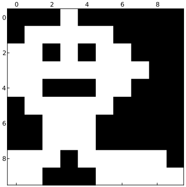
$$C_1 = \{A : A[i,:] \in \mathcal{R}_i \text{ for } i = 1, \dots, m\},$$
$$C_2 = \{A : A[:,j] \in \mathcal{C}_j \text{ for } j = 1, \dots, n\}.$$

Given an incomplete nonogram puzzle, A is a solution if and only if

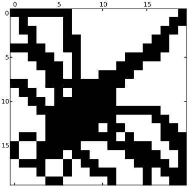
$$A \in C_1 \cap C_2.$$

Nonograms: Computational Results

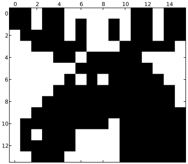
From 1000 random replications, the following nonograms were solved in every instance.



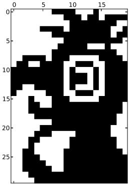
A spaceman.



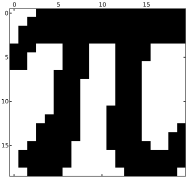
A dragonfly.



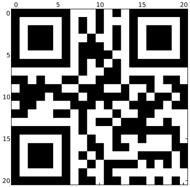
A moose.



A parrot.



The number π.



“Hello from CARMA”.

Nonograms: Computational Details

- Computing the projections onto C_1 and C_2 is difficult.
- We do not know an efficient way to do so.
  - Our approach: Pre-compute all legal cluster size sequences (slow).
- Only a few Douglas–Rachford iterations are required to solve (fast).

In contrast other problems, frequently, have relatively simple projections but require many more iterations.

This suggests the following:

Trade-off between simplicity of projection operators and the number of iterations required.

Nonograms: Computational Details

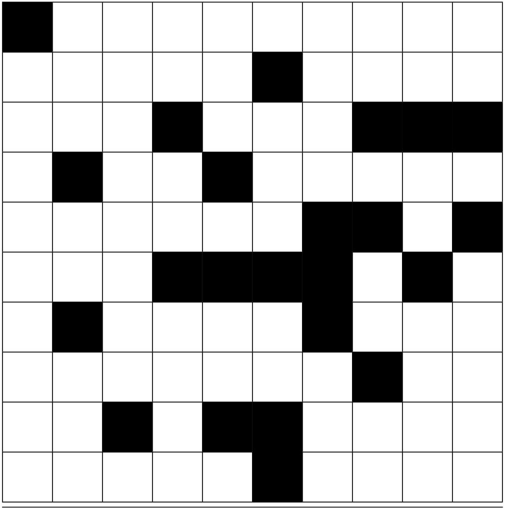
- Computing the projections onto C_1 and C_2 is difficult.
- We do not know an efficient way to do so.
    - Our approach: Pre-compute all legal cluster size sequences (slow).
- Only a few Douglas–Rachford iterations are required to solve (fast).

In contrast other problems, frequently, have relatively simple projections but require many more iterations.

This suggests the following:

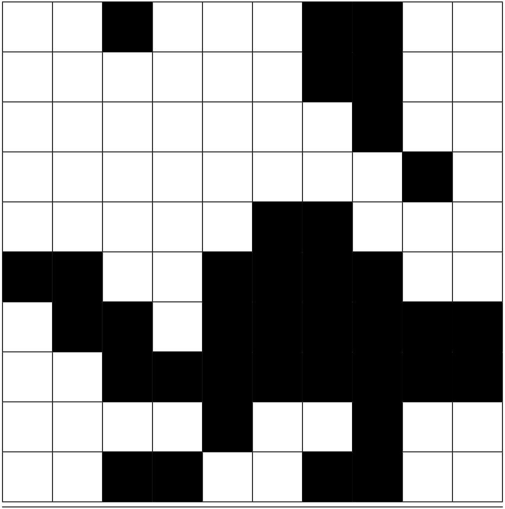
Trade-off between simplicity of projection operators and the number of iterations required.

Nonograms: An example



Iteration: 0 (random initialisation)

Nonograms: An example



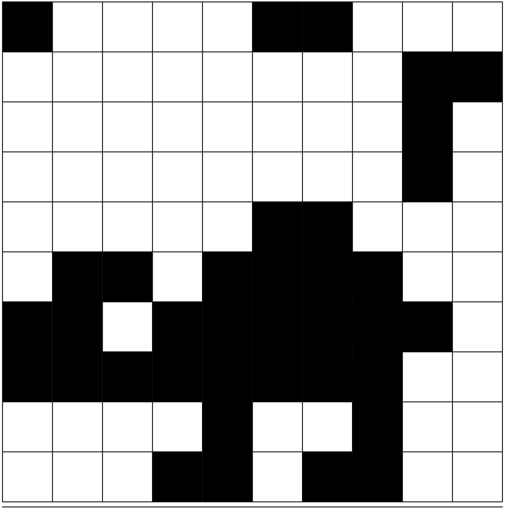
Iteration: 1

Nonograms: An example

			■					■	■
						■	■		
								■	
								■	
								■	■
■	■			■	■	■	■		
■	■		■	■	■	■	■	■	
■	■	■	■	■	■	■	■		
				■			■		
			■	■		■	■		

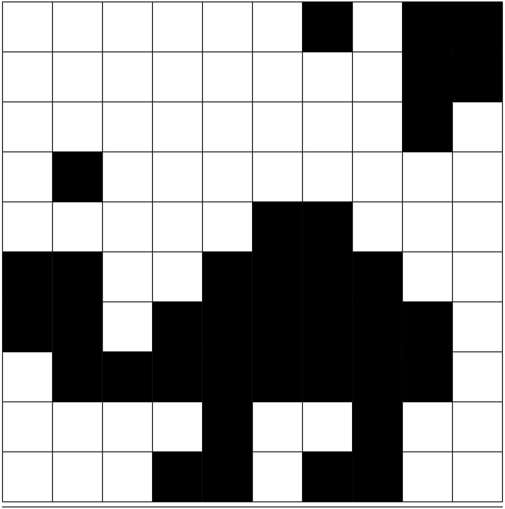
Iteration: 2

Nonograms: An example



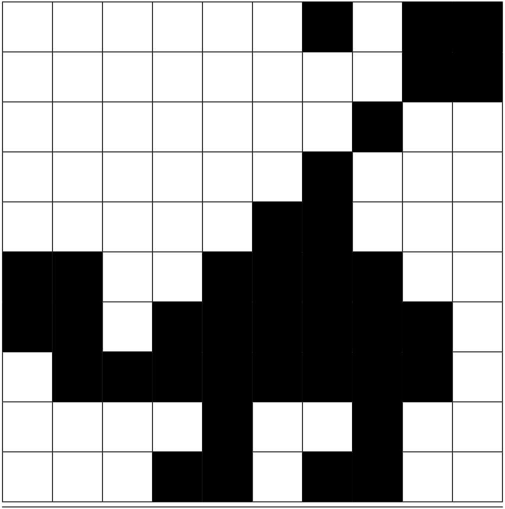
Iteration: 3

Nonograms: An example



Iteration: 4

Nonograms: An example



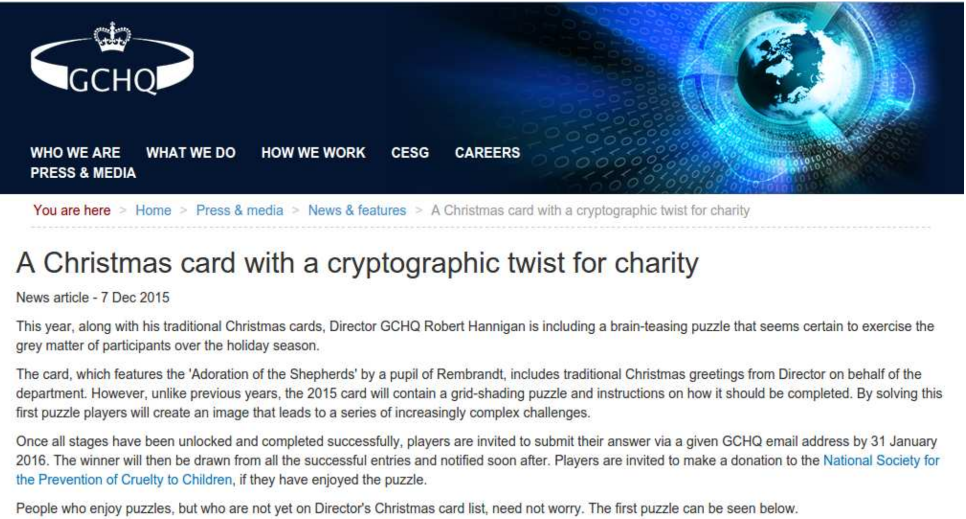
Iteration: 5

Nonograms: An example

□	□	□	□	□	□	■	□	■	■
□	□	□	□	□	□	□	□	■	■
□	□	□	□	□	□	□	■	□	□
□	□	□	□	□	□	□	□	□	■
□	□	□	□	□	■	■	□	□	□
■	■	□	□	■	■	■	■	□	□
■	■	□	■	■	■	■	■	■	□
□	■	■	■	■	■	■	■	■	□
□	□	□	□	■	□	□	■	□	□
□	□	□	■	■	□	■	■	□	□

Iteration: 6 (solved)

GCHQ's 2015 Christmas Puzzle


GCHQ
WHO WE ARE
WHAT WE DO
HOW WE WORK
CESG
CAREERS
PRESS & MEDIA
You are here > Home > Press & media > News & features > A Christmas card with a cryptographic twist for charity

A Christmas card with a cryptographic twist for charity

News article - 7 Dec 2015

This year, along with his traditional Christmas cards, Director GCHQ Robert Hannigan is including a brain-teasing puzzle that seems certain to exercise the grey matter of participants over the holiday season.

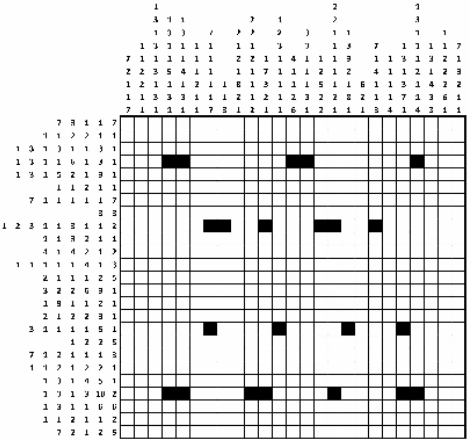
The card, which features the 'Adoration of the Shepherds' by a pupil of Rembrandt, includes traditional Christmas greetings from Director on behalf of the department. However, unlike previous years, the 2015 card will contain a grid-shading puzzle and instructions on how it should be completed. By solving this first puzzle players will create an image that leads to a series of increasingly complex challenges.

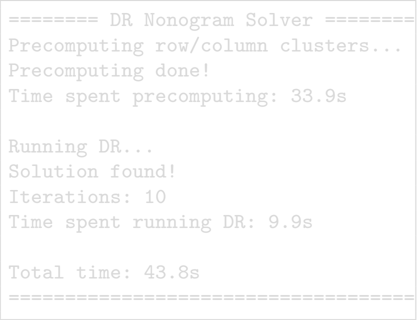
Once all stages have been unlocked and completed successfully, players are invited to submit their answer via a given GCHQ email address by 31 January 2016. The winner will then be drawn from all the successful entries and notified soon after. Players are invited to make a donation to the National Society for the Prevention of Cruelty to Children, if they have enjoyed the puzzle.

People who enjoy puzzles, but who are not yet on Director's Christmas card list, need not worry. The first puzzle can be seen below.

[5] Kudos to Veit Elser who made us aware of the puzzle.

GCHQ's 2015 Christmas Puzzle



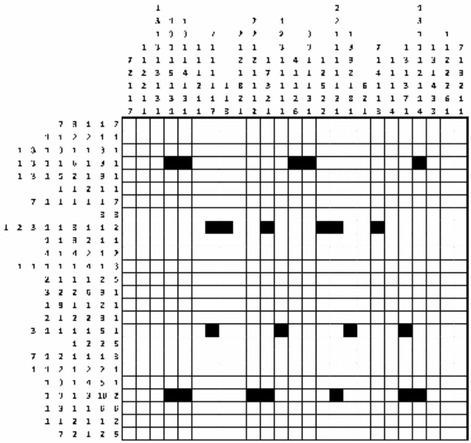


```
========= DR Nonogram Solver =========
Precomputing row/column clusters...
Precomputing done!
Time spent precomputing: 33.9s

Running DR...
Solution found!
Iterations: 10
Time spent running DR: 9.9s

Total time: 43.8s
======================================
```

GCHQ's 2015 Christmas Puzzle



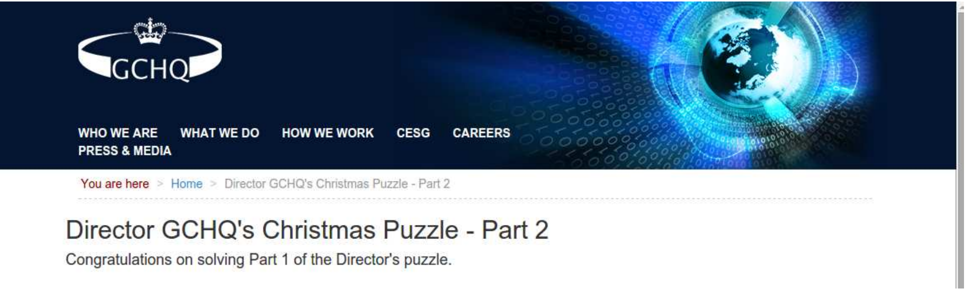
```
======== DR Nonogram Solver ========
Precomputing row/column clusters...
Precomputing done!
Time spent precomputing: 33.9s

Running DR...
Solution found!
Iterations: 10
Time spent running DR: 9.9s

Total time: 43.8s
====================================
```

GCHQ's 2015 Christmas Puzzle

The solution is a QR code which directs to the following website.

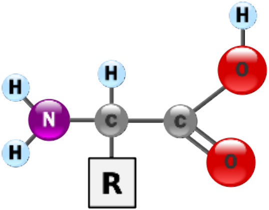

GCHQ
WHO WE ARE
WHAT WE DO
HOW WE WORK
CESG
CAREERS
PRESS & MEDIA
You are here > Home > Director GCHQ's Christmas Puzzle - Part 2

Director GCHQ's Christmas Puzzle - Part 2

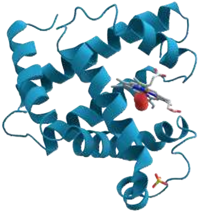
Congratulations on solving Part 1 of the Director's puzzle.

Protein Conformation Determination and EDMs

Proteins are large biomolecules comprising of multiple amino acid chains.



Generic amino acid



Myoglobin

They participate in virtually every cellular process, and knowledge of structural conformation gives insights into the mechanisms by which they perform.

Protein Conformation Determination and EDMs

One technique that can be used to determine conformation is nuclear magnetic resonance (NMR) spectroscopy. However, NMR is only able to resolve short inter-atomic distances (*i.e.*, < 6Å). For 1PTQ (404 atoms) this corresponds to $< 8\%$ of the total inter-atomic distances.

We say $D = (D_{ij}) \in \mathbb{R}^{m \times m}$ is a Euclidean distance matrix (EDM) if there exists points $p_1, \ldots, p_m \in \mathbb{R}^q$ such that

$$D_{ij} = \|p_i - p_j\|^2.$$

When this holds for points in $\mathbb{R}^q$, we say that D is embeddable in $\mathbb{R}^q$.

We formulate protein reconstruction as a matrix completion problem:

Find a EDM, embeddable in $\mathbb{R}^s$ where $s := 3$,
knowing only short inter-atomic distances.

Protein Conformation Determination and EDMs

One technique that can be used to determine conformation is nuclear magnetic resonance (NMR) spectroscopy. However, NMR is only able to resolve short inter-atomic distances (*i.e.*, < 6Å). For 1PTQ (404 atoms) this corresponds to $< 8\%$ of the total inter-atomic distances.

We say $D = (D_{ij}) \in \mathbb{R}^{m \times m}$ is a Euclidean distance matrix (EDM) if there exists points $p_1, \ldots, p_m \in \mathbb{R}^q$ such that

$$D_{ij} = \|p_i - p_j\|^2.$$

When this holds for points in $\mathbb{R}^q$, we say that D is embeddable in $\mathbb{R}^q$.

We formulate protein reconstruction as a matrix completion problem:

Find a EDM, embeddable in $\mathbb{R}^s$ *where* $s := 3$,
knowing only short inter-atomic distances.

Protein Conformation Determination and EDMs

One technique that can be used to determine conformation is nuclear magnetic resonance (NMR) spectroscopy. However, NMR is only able to resolve short inter-atomic distances (*i.e.*, < 6Å). For 1PTQ (404 atoms) this corresponds to $< 8\%$ of the total inter-atomic distances.

We say $D = (D_{ij}) \in \mathbb{R}^{m \times m}$ is a Euclidean distance matrix (EDM) if there exists points $p_1, \ldots, p_m \in \mathbb{R}^q$ such that

$$D_{ij} = \|p_i - p_j\|^2.$$

When this holds for points in $\mathbb{R}^q$, we say that D is embeddable in $\mathbb{R}^q$.

We formulate protein reconstruction as a matrix completion problem:

Find a EDM, embeddable in $\mathbb{R}^s$ where $s := 3$,
knowing only short inter-atomic distances.

A Feasibility Problem Formulation

Denote by Q the Householder matrix defined by

$$Q := I - \frac{2vv^T}{v^Tv}, \text{where } v = \left[1, 1, \ldots, 1, 1+\sqrt{m}\right]^T \in \mathbb{R}^m.$$

Theorem (Hayden–Wells 1988)

A nonnegative, symmetric, hollow matrix X, is a EDM iff $\widehat{X} \in \mathbb{R}^{(m-1)\times(m-1)}$ in

$$Q(-X)Q = \begin{bmatrix} \widehat{X} & d \\ d^T & \delta \end{bmatrix} \qquad (*)$$

is positive semi-definite (PSD). In this case, X is embeddable in $\mathbb{R}^q$ where $q = \operatorname{rank}(\widehat{X}) \leq m-1$ but not in $\mathbb{R}^{q-1}$.

Let D denote the partial EDM (obtained from NMR), and $\Omega \subset \mathbb{N} \times \mathbb{N}$ the set of indices for known entries. The problem of low-dimensional EDM reconstruction can thus be case as a feasibility problem with constraints:

$$C_1 = \{X \in \mathbb{R}^{m\times m} : X \geq 0, X_{ij} = D_{ij} \text{ for } (i,j) \in \Omega\},$$

$$C_2 = \{X \in \mathbb{R}^{m\times m} : \widehat{X} \text{ in } (*) \text{ is PSD with } \operatorname{rank} \widehat{X} \leq s := 3\}.$$

A Feasibility Problem Formulation

Recall the constraint sets:

$$C_1 = \{X \in \mathbb{R}^{m \times m} : X \geq 0, X_{ij} = D_{ij} \text{ for } (i,j) \in \Omega\},$$
$$C_2 = \{X \in \mathbb{R}^{m \times m} : \widehat{X} \text{ in } (*) \text{ is PSD with } \operatorname{rank} \widehat{X} \leq s := 3\}.$$

Now,

- C_1 is a convex set (intersection of cone and affine subspace).
- C_2 is convex iff $m \leq 2$ (in which case $C_2 = \mathbb{R}^{m \times m}$).

For interesting problems, C_2 is **never convex**.

Computing Projections and Reflections

Recall the constraint sets:

$$C_1 = \{X \in \mathbb{R}^{m\times m} : X \geq 0, X_{ij} = D_{ij} \text{ for } (i,j) \in \Omega\},$$
$$C_2 = \{X \in \mathbb{R}^{m\times m} : \widehat{X} \text{ in } (*) \text{ is PSD with } \operatorname{rank} \widehat{X} \leq s := 3\}.$$

The projection onto C_1 is given (point-wise) by

$$P_{C_1}(X)_{ij} = \begin{cases} D_{ij} & \text{if } (i,j) \in \Omega, \\ \max\{0, X_{ij}\} & \text{otherwise.} \end{cases}$$

The projection onto C_2 is the set

$$P_{C_2}(X) = \left\{ -Q \begin{bmatrix} \widehat{Y} & d \\ d^T & \delta \end{bmatrix} Q : Q(-X)Q = \begin{bmatrix} \widehat{X} & d \\ d^T & \delta \end{bmatrix}, \begin{array}{l} \widehat{X} \in \mathbb{R}^{(m-1)\times(m-1)}, \\ d \in \mathbb{R}^{m-1}, \delta \in \mathbb{R}, \end{array} \widehat{Y} \in P_{\mathcal{S}_3}\widehat{X} \right\},$$

where $\mathcal{S}_s$ is the set of PSD matrices of rank s or less.

- Computing $P_{\mathcal{S}_s}(\widehat{X})$ = spectral decomposition → threshold eigenvalues.

Computing Projections and Reflections

Recall the constraint sets:

$$C_1 = \{X \in \mathbb{R}^{m\times m} : X \geq 0, X_{ij} = D_{ij} \text{ for } (i,j) \in \Omega\},$$
$$C_2 = \{X \in \mathbb{R}^{m\times m} : \widehat{X} \text{ in } (*) \text{ is PSD with } \operatorname{rank} \widehat{X} \leq s := 3\}.$$

The projection onto C_1 is given (point-wise) by

$$P_{C_1}(X)_{ij} = \begin{cases} D_{ij} & \text{if } (i,j) \in \Omega, \\ \max\{0, X_{ij}\} & \text{otherwise.} \end{cases}$$

The projection onto C_2 is the set

$$P_{C_2}(X) = \left\{ -Q \begin{bmatrix} \widehat{Y} & d \\ d^T & \delta \end{bmatrix} Q : Q(-X)Q = \begin{bmatrix} \widehat{X} & d \\ d^T & \delta \end{bmatrix}, \begin{matrix} \widehat{X} \in \mathbb{R}^{(m-1)\times(m-1)}, \\ d \in \mathbb{R}^{m-1}, \delta \in \mathbb{R}, \end{matrix} \; \widehat{Y} \in P_{\mathcal{S}_3}\widehat{X} \right\},$$

where $\mathcal{S}_s$ is the set of PSD matrices of rank s or less.

- Computing $P_{\mathcal{S}_s}(\widehat{X})$ = spectral decomposition → threshold eigenvalues.

Computing Projections and Reflections

Recall the constraint sets:

$$C_1 = \{X \in \mathbb{R}^{m\times m} : X \geq 0, X_{ij} = D_{ij} \text{ for } (i,j) \in \Omega\},$$
$$C_2 = \{X \in \mathbb{R}^{m\times m} : \widehat{X} \text{ in } (*) \text{ is PSD with } \operatorname{rank} \widehat{X} \leq s := 3\}.$$

The projection onto C_1 is given (point-wise) by

$$P_{C_1}(X)_{ij} = \begin{cases} D_{ij} & \text{if } (i,j) \in \Omega, \\ \max\{0, X_{ij}\} & \text{otherwise.} \end{cases}$$

The projection onto C_2 is the set

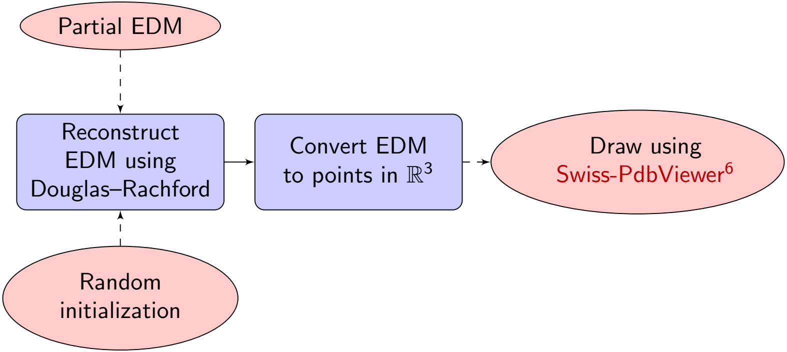
$$P_{C_2}(X) = \left\{ -Q \begin{bmatrix} \widehat{Y} & d \\ d^T & \delta \end{bmatrix} Q : Q(-X)Q = \begin{bmatrix} \widehat{X} & d \\ d^T & \delta \end{bmatrix}, \begin{array}{c} \widehat{X} \in \mathbb{R}^{(m-1)\times(m-1)}, \\ d \in \mathbb{R}^{m-1}, \delta \in \mathbb{R}, \end{array} \widehat{Y} \in P_{\mathcal{S}_3}\widehat{X} \right\},$$

where $\mathcal{S}_s$ is the set of PSD matrices of rank s or less.

- Computing $P_{\mathcal{S}_s}(\widehat{X})$ = spectral decomposition → threshold eigenvalues.

The Algorithmic Approach

The reconstruction approach can be summarised as follows:



Partial EDM

Reconstruct EDM using Douglas–Rachford

Convert EDM to points in $\mathbb{R}^3$

Draw using Swiss-PdbViewer[6]

Random initialization

[1] http://spdbv.vital-it.ch/

Experiment: Six Test Proteins

Experiment: We consider the simplest realistic protein conformation determination problem.

NMR experiments were simulated for proteins with known conformation by computing the partial EDM containing all inter-atomic distances < 6Å.

Table: Six proteins from the RCSB Protein Data Bank.[7]

Protein	# Atoms	# Residues	Known Distances
1PTQ	404	50	8.83%
1HOE	581	74	6.35%
1LFB	641	99	5.57%
1PHT	988	85	4.57%
1POA	1067	118	3.61%
1AX8	1074	146	3.54%

[2] `http://www.rcsb.org/`

Experiment: Six Test Proteins

Table: Average (worst) results: **5,000** iterations, five random initializations.

Protein	Problem Size	Rel. Error (dB)	RMS Error	Max Error
1PTQ	81,406	-83.6 (-83.7)	0.02 (0.02)	0.08 (0.09)
1HOE	168,490	-72.7 (-69.3)	0.19 (0.26)	2.88 (5.49)
1LFB	205,120	-47.6 (-45.3)	3.24 (3.53)	21.68 (24.00)
1PHT	236,328	-60.5 (-58.1)	1.03 (1.18)	12.71 (13.89)
1POA	568,711	-49.3 (-48.1)	34.09 (34.32)	81.88 (87.60)
1AX8	576,201	-46.7 (-43.5)	9.69 (10.36)	58.55 (62.65)

- The reconstructed EDM is compared to the actual EDM using:

$$\text{Relative error (decibels)} = 10\log_{10}\left(\frac{\|P_A x_n - P_B R_A x_n\|^2}{\|P_A x_n\|^2}\right).$$

- The reconstructed points in $\mathbb{R}^3$ are then compared using:

$$\text{RMS Error} = \left(\sum_{k=1}^{m} \|z_k - z_k^{\text{actual}}\|^2\right)^{1/2}, \quad \text{Max Error} = \max_{k=1,\dots,m} \|z_k - z_k^{\text{actual}}\|,$$

which are computed up to translation, reflection and rotation.

Experiment: Six Test Proteins

Table: Average (worst) results: 5,000 iterations, five random initializations.

Protein	Problem Size	Rel. Error (dB)	RMS Error	Max Error
1PTQ	81,406	-83.6 (-83.7)	0.02 (0.02)	0.08 (0.09)
1HOE	168,490	-72.7 (-69.3)	0.19 (0.26)	2.88 (5.49)
1LFB	205,120	-47.6 (-45.3)	3.24 (3.53)	21.68 (24.00)
1PHT	236,328	-60.5 (-58.1)	1.03 (1.18)	12.71 (13.89)
1POA	568,711	-49.3 (-48.1)	34.09 (34.32)	81.88 (87.60)
1AX8	576,201	-46.7 (-43.5)	9.69 (10.36)	58.55 (62.65)

- The reconstructed EDM is compared to the actual EDM using:

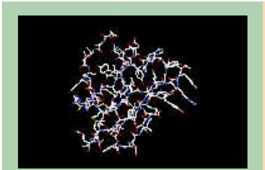
$$\text{Relative error (decibels)} = 10\log_{10}\left(\frac{\|P_A x_n - P_B R_A x_n\|^2}{\|P_A x_n\|^2}\right).$$

- The reconstructed points in $\mathbb{R}^3$ are then compared using:

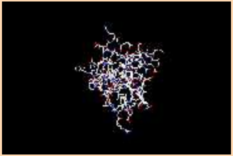
$$\text{RMS Error} = \left(\sum_{k=1}^{m} \|z_k - z_k^{\text{actual}}\|^2\right)^{1/2}, \quad \text{Max Error} = \max_{k=1,\ldots,m} \|z_k - z_k^{\text{actual}}\|,$$

which are computed up to translation, reflection and rotation.

Experiment: Six Test Proteins



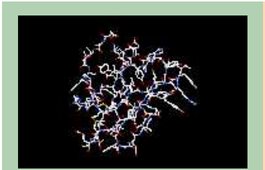
1HOE (actual)

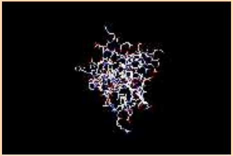


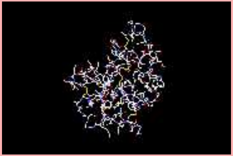
1LFB (actual)

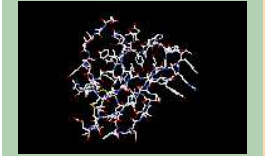
1POA (actual)

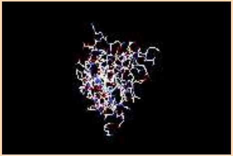
Experiment: Six Test Proteins

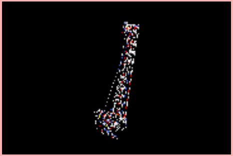

1HOE (actual)


1LFB (actual)


1POA (actual)


1HOE (-72.7dB)


1LFB (-60.5dB)


1POA (-49.3dB)

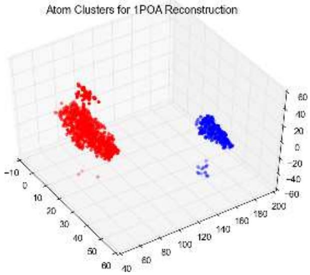
1HOE is good, 1LFB is mostly good, and 1POA has two good pieces.

Experiment: Six Test Proteins

Let's take a closer look at the bad 1POA reconstructions.

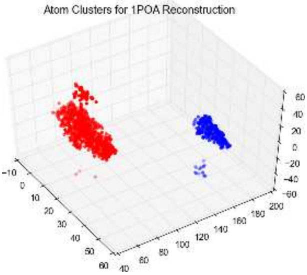
Experiment: Six Test Proteins

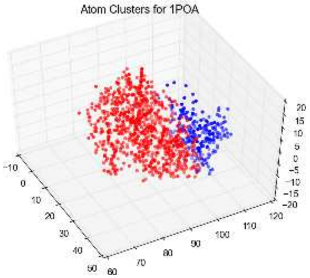
Let's take a closer look at the bad 1POA reconstructions. We *partition* the bad protein's atoms into two clusters: blue and red. We colour the same atoms in the actual structure.



Experiment: Six Test Proteins

Let's take a closer look at the bad 1POA reconstructions. We *partition* the bad protein's atoms into two clusters: blue and red. We colour the same atoms in the actual structure.





- The reconstructed protein's clusters splits actual conformation nicely in two 'halves'.

Experiment: A Better Stopping Criterion?

Optimising our implementation gave a ten-fold speed-up. We performed the following experiment:

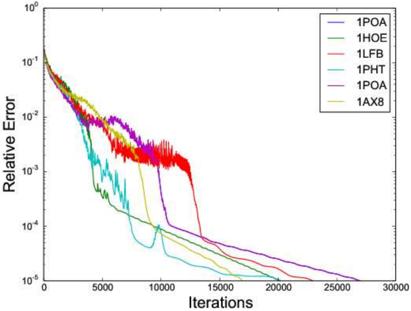
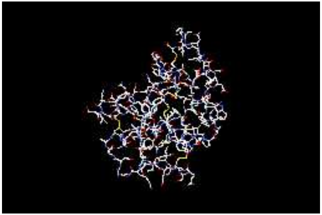


Figure: Relative error by iterations (vertical axis logarithmic).

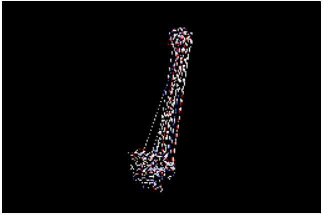
- For $< 5,000$ iterations, the error exhibits non-monotone oscillatory behaviour. It then decreases sharply. Beyond this progress is slower.
- Early termination to blame? $\longrightarrow$ Terminate when error < -100dB.

A More Robust Stopping Criterion

The "un-tuned" implementation (worst reconstruction from previous slide):



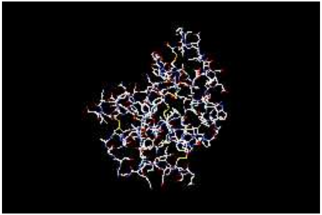
1POA (actual)



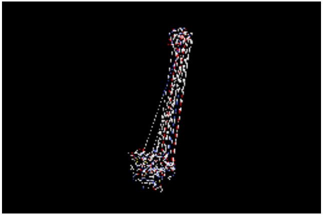
5,000 steps, -49.3dB

A More Robust Stopping Criterion

The “un-tuned” implementation (worst reconstruction from previous slide):

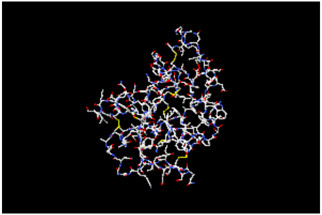


1POA (actual)

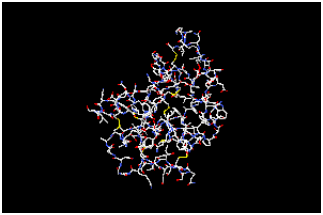


5,000 steps, -49.3dB

The optimised implementation:



1POA (actual)



28,500 steps, -100dB (perfect!)

- Similar results observed for the other test proteins.