Basis pursuit ℓ_1 minimization and its biomedical applications

Lorenzo Simone[1], **Sharathchandra Yanamandra** [2]

[1] Università di Pisa (Artificial Intelligence); l.simone3@studenti.unipi.it
[2] Università di Pisa (Artificial Intelligence); s.yanamandra@studenti.unipi.it

Version March 10, 2021 submitted to Computational Mathematics for Learning and Data Analysis

1. Introduction

2. Input data and applications

The objective of providing a dataset or more broadly a specific application for compressive sensing has played a huge role in this project. In literature amongst the major applications in biomedical field such as electroencephalografic (EEG), electrocardiogram (ECG) there is biomedical imaging [1]. We provided two applications for basis pursuit:

- reconstructing a signal through a sparse sum of cosines representation
- ECG signal compression exploiting sparsity in spectral domain

2.1. Sparse sum of cosines

The first application covers a sparse sum of cosines reconstruction of an unknown function provided a set of known measurements. Amongst the proposed applications it is the most trivial, but its attractiveness relies on the strong correlation with the foundations of Fourier analysis. The latter is made clear by Fourier theorem stating informally that a periodic function $f(x)$ may be expressed as the sum of a series of sine or cosine terms, each of which with a specific amplitude and phase coefficients, having the name of *Fourier coefficients*. In a detailed definition the problem can be stated as follows:

$$\begin{gathered}\text{given a set of ordered pairs also known as observations } \{(x_i, f(x_i)) \mid i = 1, ..., m\} \\ \text{we proceed solving the following underdetermined linear system of equations } (Ay = b)\end{gathered} \tag{1}$$

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_m) \end{bmatrix}$$

$$\begin{gathered}A \in \mathbb{R}^{m \times n} \text{ with } m \ll n \text{ and } a_{ij} = cos(jx_i), y \in \mathbb{R}^n \text{ having } \|y\|_0 = k \ll n \text{ and } b \in \mathbb{R}^m \\ \text{while minimizing } \|y\|_1 \text{ with each equality constraint summarized as } \hat{f}(x_i) = \sum_{j=1}^{n} a_{ij} * \cos(jx_i).\end{gathered} \tag{2}$$

We also investigated known signal waves which can not be expressed as a a discrete sum of cosines, so that we can judge the closeness of our proposed approximation. In order to do so we

uniformly distributed sampled points x_i over time domain and retained their respective amplitude values $f(x_i)$ according to the waves:

- Square wave [1]: $f(x) = sgn(sin(2\pi fx)) = \frac{4}{\pi} \sum_{k=1}^{\infty} \frac{sin(2\pi(2k-1)fx)}{2k-1}$
- Triangle wave: $f(x) = \frac{2a}{\pi} arcsin(sin(2\pi fx)) = \frac{8}{\pi^2} \sum_{k=1,3,5,\dots}^{\infty} sin(k\pi fx) \frac{(-1)^{\frac{k-1}{2}}}{k^2}$
- Sawtooth wave: $f(x) = -\frac{2a}{\pi} arctan(cot(x\pi f)) = \frac{1}{2} - \frac{1}{\pi} \sum_{k=1}^{\infty} \frac{1}{k} sin(k\pi fx)$

The plots over time of the analyzed waves are furtherly depicted in Figure 1.

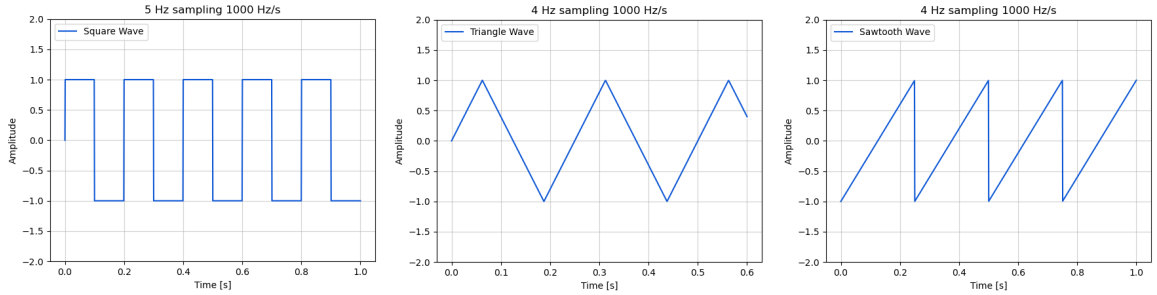
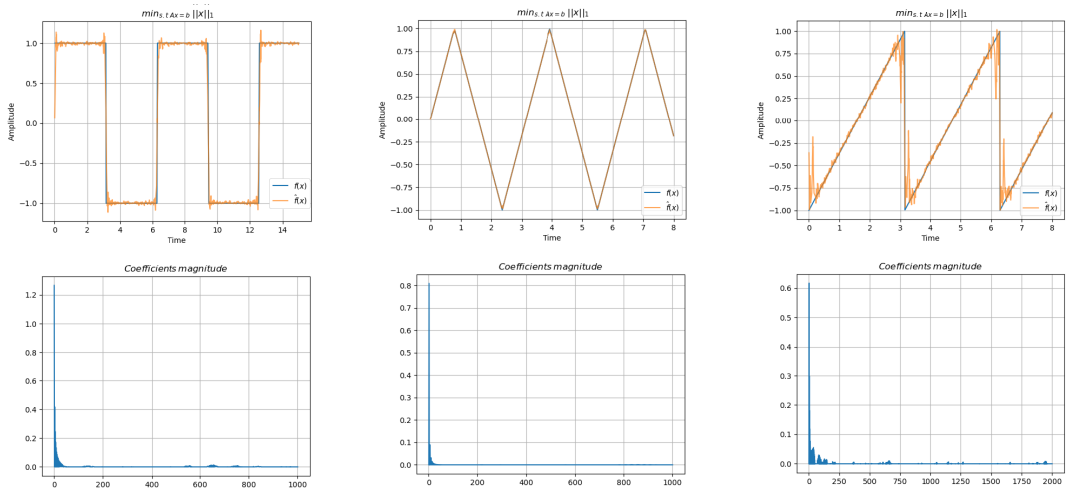


Figure 1. Amplitude plots over time of different signals used to sample observations. Firstly we have the square wave, secondly the triangular and lastly the sawtooth wave.

The sampling procedure allows to construct the A matrix and the b vector used for equality constraints. After solving the linear programming *basis pursuit* problem we can investigate the capability of the sum of cosines reconstruction to approximate the original function used for sampling in Section 4.1.



(a) **(b)** **(c)**

Figure 2. The first row depicts the overlapping of the original signal in blue $f(x)$ used for sampling and the reconstructed sum of cosines approximation $\hat{f}(x)$. The sparsity of the solution coefficients can be visualized by the second row of plots. (**a**) Square wave, (**b**) Triangle wave, (**c**) Sawtooth wave,

[1] For each wave we provide the discrete amplitude function as long as the Fourier series equivalent formulation

2.2. ECG signal compression

In this application we utilize ECG signals by undersampling the original signal given a percentage of compression and reconstructing it by using compressed sensing. The validity of compressed sensing in this application is based on the fact that the signal in the frequency domain is highly sparse. Amongst all the configurations, we are interested in the simplest and most sparse solution according to the ℓ_1-norm matching exactly or as closely as possible the known data.
In Figure 3 we show on the left a sample of an ECG signal sampled at 1kHz for 5 seconds with its Discrete Cosine Transform (DCT), we preferred using the latter over Discrete Fourier Transform (DFT) becausing having to deal with real sparse coefficients was more efficient than having complex ones.

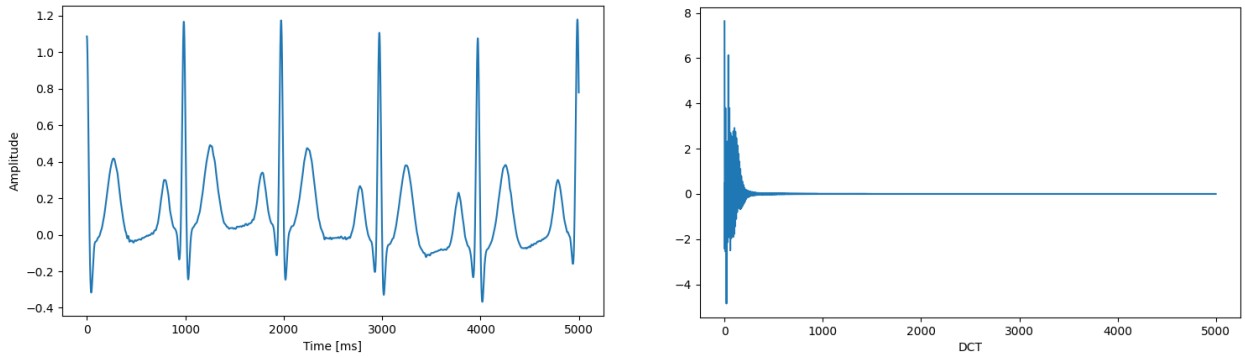


Figure 3. (a) ECG signal sampled at 1kHz for 5 seconds having an average of 60 beats per minute, **(b)** Discrete cosine transform of the input ECG signal representing a highly sparse basis.

As a recap we are still solving the task:

$$\min_{x\ s.t.\ Ax=b} \|x\|_1 \tag{3}$$

Specifically, in the context of this application the vector b contains the data samples in the temporal domain and the solution x belongs to the frequency domain, the role of the short fat matrix $A \in \mathbb{R}^{m\times n}$ is to perform both sampling and transformation from frequency to temporal domain.
In order to construct the matrix A we followed a well established methodology [2,3]. In our application the ECG signal can be regarded as a vector f being a linear combination of the DCT expressed as ψ.

$$f = \psi c \tag{4}$$

The sampling procedure consists in a linear operator ϕ

$$b = \phi f \tag{5}$$

From the abov$\times 10^{mentionedequationsit\,follows that}$A = $\phi\psi$, and it is made of the rows sampled from the domain transform matrix ψ, being the Inverse Discrete Cosine Trasnform (IDCT) applied to the columns of the identity matrix. A visual representation of the matrix A applied to this task is depicted in Figure 4, having dimensions 1000×5000, being respectively the sampling rate (1kHz) and the elapsed time (5000 ms).

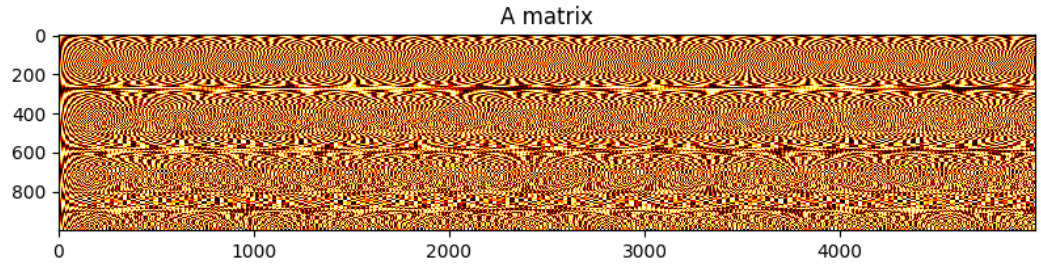


Figure 4. Visual representation of the matrix A applied to the ECG signal wave depicted in Figure 3, having dimensions 1000×5000, being respectively the sampling rate (1kHz) and the elapsed time (5000 ms)

Once the building blocks of the original problem are constructed we solved it and converted the solution from the frequency to the temporal domain. We have investigated different percentages of compression rate, which is involved in the random sampling process, obtaining satisfying results even with only 10% of the original signal data.

We provide an example of the overall process: starting from the ECG signal depicted in Figure 3 we uniformly random sampled 10% of it and we constructed the A matrix, obtaining the solution to the classical basis pursuit problem which was finally transformed in the temporal domain. The second row from Figure 5, qualitatively describes the result of the reconstruction using the above described process and settings for the compression rate.

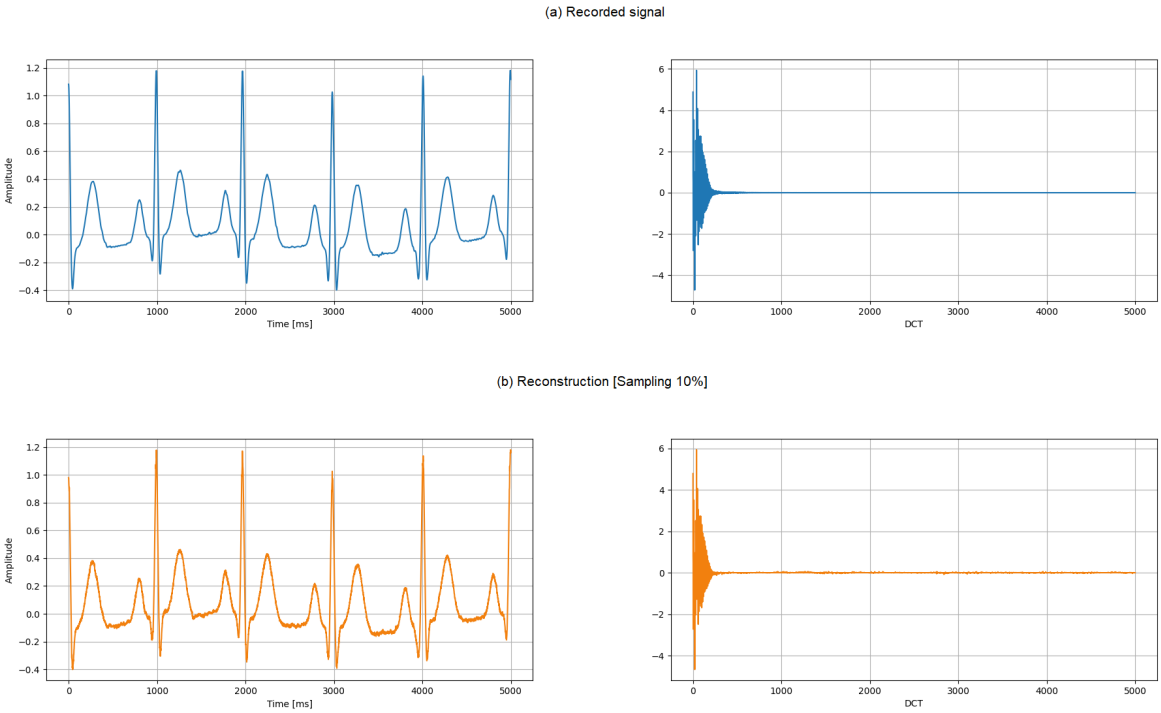


Figure 5. (a) Original ECG signal sampled at 1kHz for 5 seconds having an average of 60 beats per minute next to its DCT, **(b)** Reconstructed ECG signal from random sampling only 10% of the original wave next to its DCT representing the solution.

3. Basis Pursuit as Linear Programming

The original Basis Pursuit (BP) is a convex optimization problem formulated as:

$$\min_{x\ s.t.\ Ax=b} \|x\|_2\,, \tag{6}$$

the assumption of convexity for this problem holds from the two basic properties for the set on which we optimize $\mathbb{R}^n$ and for the objective function:

1. **Convex set:**
   Any convex combination of two vectors in $\mathbb{R}^n$, does still belong in $\mathbb{R}^n$.

   Let x,y $\in \mathbb{R}^n$ and $\alpha \in [0,1]$
   $\alpha x + (1-\alpha)y = \langle \alpha x_k : k \in \mathbb{N}\rangle + \langle (1-\alpha)y_k : k \in \mathbb{N}\rangle = \langle \alpha x_k + (1-\alpha)y_k : k \in \mathbb{N}\rangle = z \in \mathbb{R}^n$

2. **Convex objective function**
   From basic properties of every $p-$norm and using triangle inequality we can verify that our objective function $\|x\|_1$ is convex:

   $f : V \to \mathbb{R}$ is convex $\Leftrightarrow \forall v, w \in V, \lambda \in [0,1] : f(\lambda v + (1-\lambda)w) \leq \lambda f(v) + (1-\lambda)f(w)$, which applied to ℓ_1 and norms in general gets straightforwardly:

   $$\|\lambda v + (1-\lambda)w\|_1 \leq \|\lambda v\|_1 + \|(1-\lambda)w\|_1 = \lambda \|v\|_1 + (1-\lambda)\|w\|_1.$$

3.1. LP equivalent formulation

We proceed by reformulating the original problem as a linear programming problem having an equivalent form and a well-defined constant gradient $\nabla f(v) = \mathbf{1}^{2n}$:

$$x = v^+ - v^-,\ \ v_i^+ = \max\{x_i, 0\},\ \ v_i^- = \max\{-x_i, 0\} \tag{7}$$

Then Basis Pursuit (6) can be also written like:

$$\begin{aligned} &\min_v \textstyle\sum_{i=1}^n v_i^+ + v_i^- \\ &\\ &\text{subject to } A(v^+ - v^-) = b, \\ &v^+ \succeq \mathbf{0} \\ &v^- \succeq \mathbf{0} \end{aligned} \tag{8}$$

3.2. Asymmetric dual pair formulation

Usually, in order to find an optimal value for the objective function for an LP we make reference to the dual of the asymmetric pair, being another LP problem having many syntactical and non properties in common with the first. We proceed by defining the three main concepts behind our *revised simplex* implementation, in order to solve the *basis pursuit*.

First of all we define the primal (P) and the dual (D) from the standard asymmetric pair formulation:

$$(P)\quad \max\{cx : Ax \leq b\} \qquad\qquad (D)\quad \min\{yb : yA = c, y \geq 0\} \tag{9}$$

We also provide the new matrix $A' \in \mathbb{R}^{m\times 2n}$ and the solution vector $v \in \mathbb{R}^{2n}$ based on the reformulation from (8), this requires a change of sign for the respective columns.

$$A' = \begin{bmatrix} A & -A \end{bmatrix} \qquad v = \begin{bmatrix} v^+ \\ v_- \end{bmatrix} \tag{10}$$

$$(P) \quad \begin{array}{lllll} \max & b_1 x_1 & +b_2 x_2 & +\ldots+ & b_m x_m \\ \text{subject to} & A_{11}^T x_1 & +A_{12}^T x_2 & +\ldots+ & A_{1m}^T x_m \leq c_1 \\ & & & \vdots & \\ & A_{2n,1}^T x_1 & +A_{2n,2}^T x_2 & +\ldots+ & A_{2n,m}^T x_m \leq c_{2n} \end{array}$$

$$(D) \quad \begin{array}{lllllll} \min & c_1 v_1^+ & +c_2 v_2^+ & +\ldots+ & c_{2n} v_n^- & & \\ \text{subject to} & A_{11} v_1^+ & +A_{12} v_2^+ & +\ldots- & A_{1,2n-1} v_{n-1}^- & -A_{1,2n} v_n^- & = b_1 \\ & & & \vdots & & & \\ & A_{m,1} v_1^+ & +A_{m,2} v_2^+ & +\ldots- & A_{m,2n-1} v_{n-1}^- & -A_{m,2n} v_n^- & = b_m \\ & v_1^+, & v_2^+, & \ldots & v_{n-1}^-, & v_n^-, & \geq 0 \end{array}$$

We can notice from both the above formulations, that neither the primal (P) nor the dual (D) exhibit an admissible basis for starting the implementation of the *revised dual simplex* algorithm. Our objective from this phase is retrieving an admissible starting basis B, being a projection vector exploiting the sparsity of the A matrix and its resulting nonsingular submatrix A_B.

We proceed by establishing a working method for finding a starting basis B applied to our applications described in Section 2.2 and Section 2.1, relying on a foundamental theorem stating that the dual of the dual is the primal and by applying the two-phase method, also usually known as Big-M method [4]. An example of the latter applied to our problem is described in eq. 11.

$$(P') \quad \begin{array}{llllll} \max & b_1 x_1 & +b_2 x_2 & +\ldots+ & b_m x_m & \\ \text{subject to} & A_{11}^T x_1 & +A_{12}^T x_2 & +\ldots+ & A_{1m}^T x_m & \leq c_1 \\ & & & \vdots & & \\ & A_{2n,1}^T x_1 & +A_{2n,2}^T x_2 & +\ldots+ & A_{2n,m}^T x_m & \leq c_{2n} \\ & & & & & \\ & x_1 & \ldots & 0 & & \leq M \\ & \vdots & \ddots & & & \\ & 0 & & +x_m & & \leq M \end{array} \tag{11}$$

$$(D') \quad \begin{array}{llllllll} \min & c_1 v_1^+ & +c_2 v_2^+ & +\ldots+ & c_{2n} v_n^- & +M w_1 & +\ldots+ & M w_m \\ \text{subject to} & A_{11} v_1^+ & +A_{12} v_2^+ & +\ldots- & A_{1,2n} v_n^- & w_1 & +0 & = b_1 \\ & & & \vdots & & \vdots \quad \ddots & & \\ & A_{m,1} v_1^+ & +A_{m,2} v_2^+ & +\ldots- & A_{m,2n} v_n^- & 0 & w_m & = b_m \\ & v_1^+, & \ldots & v_{n-1}^-, & w_1, & \ldots & w_m & \geq 0 \end{array}$$

By choosing M as a vector having positive and sufficiently big components, (D') is actually equivalent to the formulation of (D), the original dual. In detail, since the dual is bounded below, also the auxiliary form of the dual will share the same property [5].

The starting point of the algorithm is now represented by the artificial basis $B = \{2n+1, ..., 2n+1+m\}$, building the matrix $A_B = I$ and having an admissible starting solution for the dual.

3.3. Linear algebra: Forrest-Tomlin update

We have to solve linear systems involving the square matrix A_B for each iteration, but we never calculate the inverse basis matrix explicitly just to solve these systems. Instead, we maintain an LU factorization for 30 iterations, by using triangular substitutions to recover the intermediate solutions.

These assumptions are based on the fact that it would be expensive to recompute the LU factorization at each step, since A_B is changing only by a single row between iterations. Actually, we could have even used a pivoting strategy based on the assumption that for some applications of BP, the A matrix is usually sparse and having specific symmetric properties, but this is left as a further optimization [6].

For instance starting from an LU factorization of A_B, with rows and columns already permuted:

$$LU = A_B \tag{12}$$

the system $A_B\bar{x} = b_B$ can be solved by the following two-steps easier procedures:

$$L\lambda = A_B, \qquad U\bar{x} = \lambda, \tag{13}$$

similarly all the systems in the form $yA = c$ are solved in the same manner, just considering that they can be rearranged in $A^T y^T = c^T$.

Straightaway, we discuss the procedure for updating the triangular matrices L and U at each step, excluding the ones in which we refactor the full LU (first iteration and multiples of 30). As we have already stated above, the only change in the matrix A_B is the row A_k substituting A_h in the basis since $h \in B$ and $k \in N$. From now on we will refer to this operation as a column update, since it would be the same by considering A_B^T instead of A_B. We will refer to the updated matrix as A_B^+.

Now, by rewriting (12) we end up with $U = L^{-1}A_B$ and $L^{-1}A_B^+$ is still an upper triangular matrix except in column h, which was changed (6.a). With the objective of moving the non-upper triangular part of the matrix to the last row, we can apply cyclic permutations shifting column h to the last column (6.b). The latter is straightforward, since we can start from the original vector of the columns of A_B^+ being $c = [1, 2, ..., h, .., n]$ and compute the permutation vector as $\pi = [1, 2, ..., n-h+2, ..., n, h]$ which applied to the identity matrix will get P_1, the row-shifting matrix and P_1^T the related column-shifting one. We propose a graphical representation of this update process in Figure 6, by using the MATLAB *spy* utility plotting function for non-zero patterns in a matrix applied to one of our iterations of the running proposed algorithm.

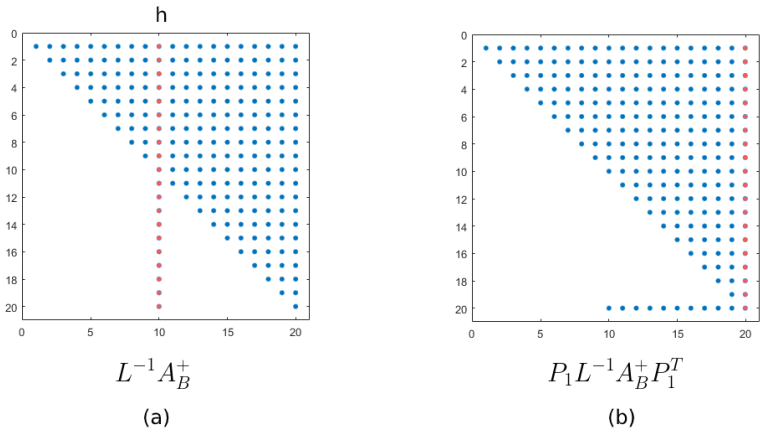


Figure 6. (a) The index for the updated column in the matrix A_B^+ is h, resulting in a pattern having a single column away from an upper triangular form. **(b)** After applying the permutation for both rows and columns respectively multiplying by P_1 and P_1^T, we end up in a situation in which only the last row is full of non-zeros.

From the matrix $P_1L^{-1}A_B^+P_1^T$ we can retrieve $P_1L^{-1}A_B^+P_1^T = L_1U_1$ by sparse Gaussian eliminations, where L_1 is different from the identity only in the last row and also U_1 is identical to $P_1L^{-1}A_B^+P_1^T$ except for the last row. Finally we can calculate the updated factorization for A_B as:

$$A_B^+ = L^+U^+, \qquad \text{where} \quad L^+ = LP_1^TL_1 \quad \text{and} \quad U^+ = U_1P_1 \tag{14}$$

3.4. Proposed algorithm pseudocode

Algorithm 1: *Revised dual simplex (Forrest-Tomlin update, Bland's anticycle rule)*

Input: **A** coefficient matrix
b vector
c cost vector
B admissible basis
maxIter allowed maximum iterations
tol_set primal empty set tolerance
tol_opt optimal solution tolerance
verb verbose mode flag

Output: $\bar{x}$ vector
$\bar{y}$ vector
cost_vector objective function value over iterations

j = 1;
refactor = 30;
while $j <$ *maxIter* **do**
    // Perform LU for the first time on A_B or after #*refactor* iterations
    if j % *refactor* == *0* **then**
        [L, U] = lu(A_B);
    else
        // Forrest-Tomlin update of LU factorization
        $\pi_1 = [1, ..., n-2, n-1, h]$
        $P_1 = I([\pi_1], :)$;
        $[L_1, U_1] = lu(P_1 L^{-1} B P_1^T)$;
        $L = L P_1^T L_1$;
        $U = U_1 P_1$;
    // Solving $x = A_b^{-1} b_b$ by LU factorization
    d = solve(L,b_b);
    $\bar{x}$ = solve(U,d);
    // Solving $\bar{y_B} = c A_b^{-1}$ by LU factorization
    $\bar{y}$ = [0,0,...,0]^m;
    λ = solve(L,c^T);
    $\bar{y_B}$ = solve(U,λ);
    // Optimal solution check
    if $A_N \bar{x} + tol\_opt \leq b_N$ **then**
        return 'Optimal solution found'
    v = solve(L,A_k^T);
    $\eta_B = solve(U, v)$;
    // Primal set emptiness check
    if $\eta_B \leq tol\_set$ **then**
        return *'Primal set is empty'*
    // Bland's anticycle rule
    $k = N(find(A_n x > b_N, 1, 'first'))$;
    $h = B(find(y/\eta \; and \; \eta > 0, 1, 'first'))$;
    // Basis update
    B(B==h) = k;
    N(N==k) = h;
    j++;

4. Experimental results

4.1. Sum of Cosines reconstruction

The first application covers a sparse sum of cosines reconstruction of unknown functions provided a set of known measurements. As a preliminary step, we tested the algorithm for functions which could be perfectly reconstructed, in detail, we used the following functions for generating the points:

- $f(x) = \cos(5x) + \cos(100x)$
- $f(x) = \cos(5x)$

Throughout each of the applications covered in this section we managed the tricky problem of tuning tolerances and stopping conditions for the method. The algorithm includes two stopping conditions referring respectively to the primal set being empty and to an optimality check. We experienced different values ranging from 10^{-8} to 10^{-5} not noticing a substancial difference in the outcome. Finally, we stuck to the value of the primal empty set check being to 10^{-6} and 10^{-7} for optimal condition.

In order to test and measure the accuracy of the reconstruction compared to the original signal, we have used the Mean Squared Error (MSE) and the Pearson correlation coefficient (ρ), defined as:

$$MSE(A,B) = \frac{1}{n}\sum_{i=1}^{n}(A_i - B_i)^2 \qquad \rho(A,B) = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{A_i - \mu_A}{\sigma_A}\right)\left(\frac{B_i - \mu_B}{\sigma_B}\right) \tag{15}$$

We ran the algorithm testing it with various combinations of the number of variables (n) and number of observations (m). For these specific trivial instances, the exact solution is perfectly recovered in few iterations, obtaining an error smaller than the machine epsilon. Based on the latter, we simply report the best results obtained with a single combination of parameters in Table 1.

$f(x)$	n	m	$\|Ax-b\|_2$	$\|x\|_1$	MSE	ρ
$\cos(5x)+\cos(100x)$	500	100	2.246437×10^{-15}	2	5.643129×10^{-32}	1
$\cos(5x)$	500	100	4.787164×10^{-15}	1	2.240498×10^{-31}	1

Table 1. Parameters and metrics for the sum of cosines reconstruction task, using trivial finite sums of cosines class of functions.

We reported a plot showing the overlapping amongst the reconstructed and the original signal in Figure 7. Since we have an equal contribution from cosine function having different frequencies, the objective function $\|x\|_1$ details which frequency component is involved in the reconstruction of the original wave.

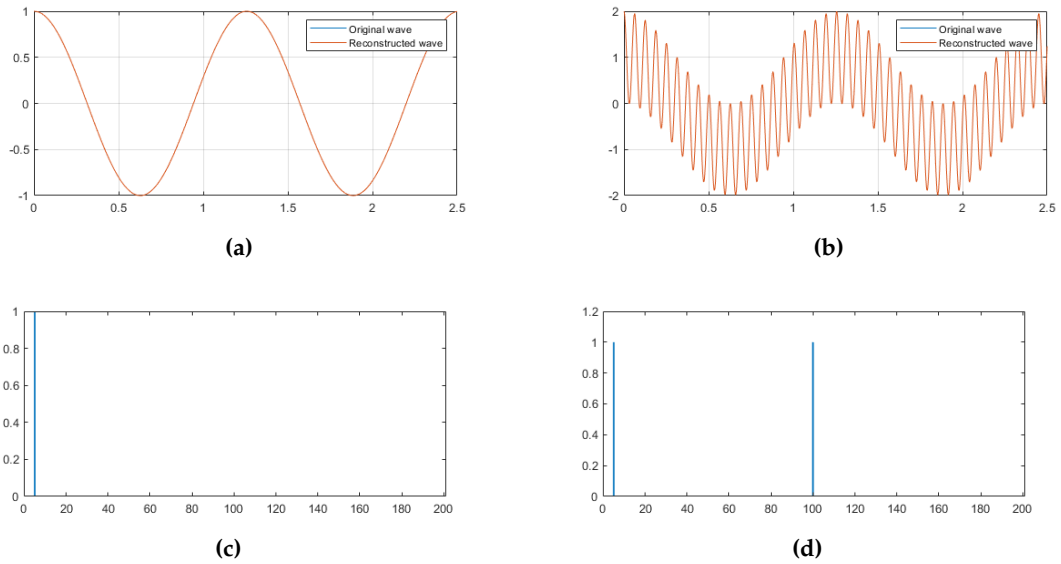


Figure 7. (**a**) Plot overlapping the original signal $f(x) = \cos(5x)$ used for sampling and the reconstructed sum of cosines approximation; (**b**) Overlap plot for $f(x) = \cos(5x) + \cos(100x)$. (**c,d**) Histogram visualizing the sparsity of the solution coefficients for both the solution vectors.

In Figure 7.c a single peak on the 5th component is noticeable, while in Figure 7.d two peaks emerge on the 5th and 100th component of the solution vector x describing which sine frequency is involved in reconstructing the signal.

Further experimental results are provided for signals unexpressable using a finite sum of cosine or sine functions. We have used the square, triangle and sawtooth waves already presented in starting sections. The quality of the solution in terms of satisfying the constraints has been verified by computing the distance $\|Ax - b\|_2$.

n	m	$\|Ax-b\|_2$	$\|x\|_1$	MSE	ρ	t(s)	Iterations
			Square wave				
1000	**500**	$\mathbf{6.184049 \times 10^{-12}}$	$\mathbf{1.191508 \times 10^{1}}$	$\mathbf{4.759236 \times 10^{-2}}$	**0.994039**	2.635641	1668
1000	200	1.336977×10^{-12}	1.030814×10^{1}	1.375599×10^{-1}	0.983268	1.579137	747
1000	100	9.115071×10^{-13}	8.962806	5.639264×10^{-1}	0.928024	0.703848	405
			Triangle wave				
1000	**300**	$\mathbf{4.855598 \times 10^{-12}}$	**2.785807**	$\mathbf{4.984034 \times 10^{-5}}$	**0.999982**	2.367487	917
1000	150	1.314703×10^{-13}	2.758031	3.348301×10^{-4}	0.999890	1.172209	537
1000	50	3.862848×10^{-14}	2.602165	5.337773×10^{-3}	0.998155	0.242113	183
			Sawtooth wave				
1000	**300**	$\mathbf{1.541872 \times 10^{-12}}$	**9.820195**	$\mathbf{3.793370 \times 10^{-2}}$	**0.985887**	2.101900	972
1000	200	8.201787×10^{-13}	8.326592	1.615636×10^{-1}	0.938157	1.679192	712
1000	100	9.999465×10^{-14}	5.401452	4.949854×10^{-1}	0.797561	0.745310	409

Table 2. Parameters and metrics for the sum of cosines reconstruction task, using trivial finite sums of cosines class of functions.

The top results in terms of MSE and ρ, obtained by using the best trade-off amongst number of variables and number of observations can be visualized in Figure 8.

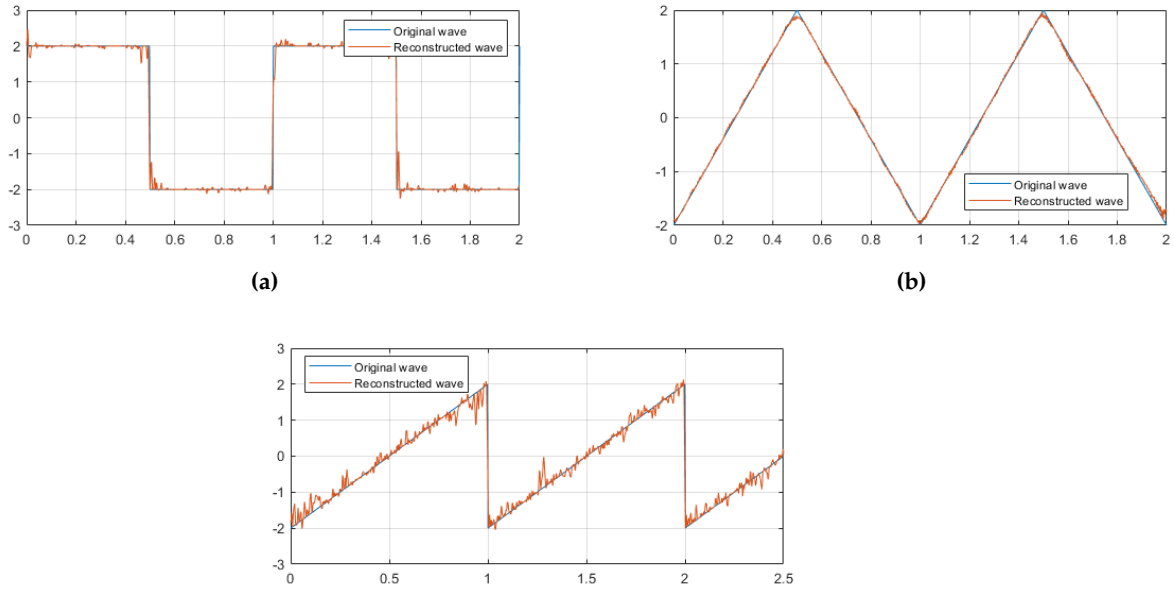


Figure 8. Overlapping of the original signal in blue $f(x)$ used for sampling and the reconstructed sum of cosines approximation $\hat{f}(x)$, (**a**) Square wave using n=1000 and m = 500, (**b**) Triangle wave using n=1000 and m = 300, (**c**) Sawtooth wave using n=1000 and m = 300.

Despite the good results shown in Table 2, from our approach, a comparison is a required step in order to assess the correctness and efficiency of the proposed solution. As from our previous knowledge on simplex which is a bound solver in linear programming will find the optimal result in much more longer time when compared with IPM solver from Matlab. Hence, the iterations are a bit longer but the requirements on the constraints and distance metrics are more precise.

The quicker algorithm (Interior point method) from table A1 results in having a lower number of iterations and could reach optimal condition by moving inside the feasible region trying to pass as many corners on the boundary. In the case of IPM, the iterations are relatively less than simplex methods as it is independent with the size of the problem.

When we compare both the approaches keenly, we observe the triangle wave and the sawtooth wave have been reasonably good for both dual and Interior point methods. However if the IPM is faster, the dual simplex results for MSE and ρ are far better.

4.2. ECG reconstruction

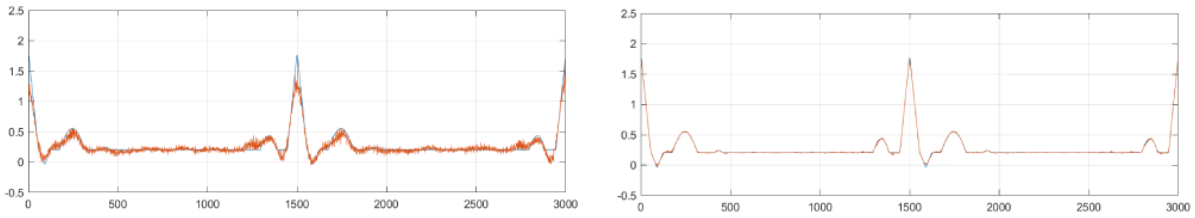
The results for the second application cover different ECG signals sampled at 0.7 kHz for 3 seconds ranging from 3 different set of heart-beating conditions:

- Bradycardia: signals lower than 60 BPM
- Tachycardia: signals above 100 BPM
- Sinus rhythm: remaining average signals

Each signal utilized in this section have been generated by a Matlab Toolbox for ECG simulation, updating the original code in order to accept different parameters such as timing, frequency, and beats per minute (BPM). The objective of this specific application is that of reconstructing an original signal sampled at a certain frequency over time, considering partial observations according to a pre-defined Compression Ratio (CR).

In Figure 9, we provide the resulting overlap amongst the reconstructed and a signal sampled at 0.7kHz for 3 seconds having 40BPM with two different CR (90% on the left, 80% on the right). The

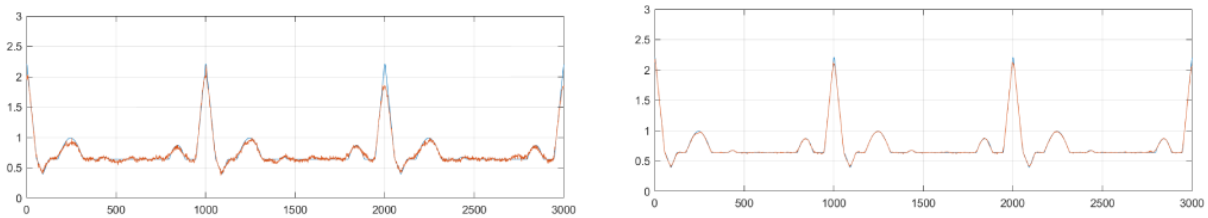
result obtained by using 90% CR, appears noisy and needs more observations, also due to the nature of the signal having a low number of characterizing peaks. Instead, the signal in Figure 9.b, by utilizing 20% of the original wave gets a reconstruction without visually appreciable differences.



(a) **(b)**

Figure 9. ECG signal sampled at 0.7kHz for 3 seconds having an average of 40 beats per minute showing bradycardia, (**a**) Overlapping the original and the reconstructed signal using 80% CR. (**b**) Overlapping the original and the reconstructed signal using 90% CR.

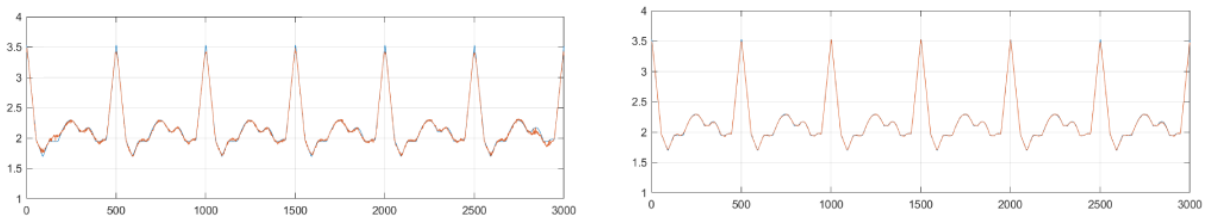
By using the same procedure, in Figure 10, we show a different signal having around 60BPM. Apparently, each signal belonging to this class show good results even by using only 10% of the original observation. We should also mention that, the sparsity assumption for compressive sensing is guaranteed by applying DCT to the signal as shown in introductory sections.



(a) **(b)**

Figure 10. ECG signal sampled at 0.7kHz for 3 seconds having an average of 60 beats per minute having sinus rhythm, (**a**) Overlapping the original and the reconstructed signal using 80% CR. (**b**) Overlapping the original and the reconstructed signal using 90% CR.

Lastly we analyze a signal sampled at the same frequency rate, but having 120BPM. The high number of peaks plays a huge role in identifying a good reconstruction even by using 80% CR. In Figure 11 we can appreciate the overall good qualitative quality of each of the two experiments.



(a) **(b)**

Figure 11. ECG signal sampled at 0.7kHz for 3 seconds having an average of 120 beats per minute showing tachycardia, (**a**) Overlapping the original and the reconstructed signal using 80% CR. (**b**) Overlapping the original and the reconstructed signal using 90% CR.

As we did previously for different waves we provide an extensive table showing a subset with the most meaningful experiments ran by measuring MSE and correlation coefficient amongst the original

and the reconstructed one by using a specific compression ratio (Table 3). Expectedly by decreasing the compression ratio which is involved in increasing the number of observations 'm', the quality of the reconstruction increases as a result. The solutions found appear to be adequately sparse according to the L-1 norm of the solution vector which is always at least one order of magnitude smaller than the original number of variables 'n'.

The notations x and x^* stand respectively for the original DCT vector and the solution obtained by the dual. In order to ensure correctness of the objective value we compared the accuracy of our optimal value compared to the target, and computing the distance $\|x^* - x\|_2$, reported in the second last column of Table 3. A visual representation of both the vectors is shown in the second column of the first and second rows in Figure A1.

BPM	n	m	$\|x^*\|_1$	MSE	ρ	$\|x^* - x\|_2$	CR
40	2100	750	92.456816	7.0×10^{-6}	0.999947	0.249430	75%
40	2100	600	91.442043	1.09×10^{-4}	0.999247	0.252230	80%
40	2100	300	82.738883	3.905×10^{-3}	0.975843	0.234737	90%
120	2100	750	190.433072	8.0×10^{-6}	0.999975	9.456573	75%
120	2100	600	189.810936	4.8×10^{-5}	0.999841	9.458531	80%
120	2100	300	185.153483	7.24×10^{-4}	0.997722	9.435898	90%
70	2100	750	120.201399	4.8×10^{-5}	0.999760	2.089961	75%
70	2100	600	119.479618	1.36×10^{-4}	0.999354	2.088543	80%
70	2100	300	114.867033	2.203×10^{-3}	0.990945	2.080364	90%

Table 3. Different signals have been sampled at 0.7kHz for 3 seconds describing different heart-beating conditions (40, 60, 90 BPM) and different compression rates (75%, 80%, 90%). We reported the dimensions of the problem as long as the objective function value and two different distances for signals: Mean Squared Error (MSE) and correlation coefficient (ρ).

4.3. Forrest Tomlin experiments

The naive approach for revised dual simplex method initially performs a matrix inversion and subsequently solves three systems for each iteration. The latter has an approximate cost of $O(n^3)$ for the matrix inversion and three times $O(n^3)$ for the systems to be solved.

Several approaches exist for exploiting a persistent rank-1 update of a matrix over several iterations. Specifically, the FT update relies on computing a new LU factorization once in a while (according to the number of iterations considered for matrix refactor) having an approximate cost of $O(n^3)$. Now, the three different systems can be easily solved with the LU updated representation by having a reduced approximate cost of $O(n^2)$ times three.

In Figure 12 we provide a graphical representation of the time of convergence for a single test iteration considering both the standard inverse approach and the FT. Along the y-axis we reported the average time for each iteration and on the x-axis the growing factor *'m'*, representing matrix size. As we can see from the plot, there is an average improvement of one order of magnitude for the second approach over the first which suggests his suitability for the method. Further investigations can be run for assessing matrix stability and other useful properties for an LP solver.

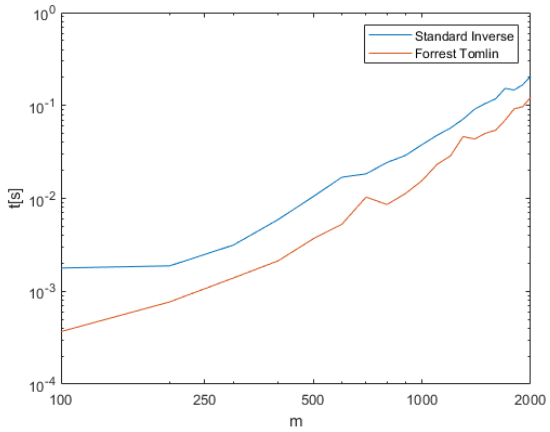


Figure 12. Single iteration time of convergence line chart for the standard inverse approach in revised dual simplex and undergoing a FT update procedure. The Y-axis represents the time elapsed in seconds on a logarithmic scale and the x-axis grows in terms of matrix size.

Appendix A

Table A1. Parameters and metrics for the sum of cosines reconstruction task using Interior Point Method from Matlab.

n	m	$\|Ax-b\|_2$	$\|x\|_1$	MSE	ρ	t(s)	Iterations
			Square wave				
1000	500	3.050674e-13	3.341861e+01	1.108982e+00	0.862073	0.235785	13
1000	300	1.150347e-08	1.832515e+01	7.908519e-01	0.895906	0.073733	15
1000	200	1.476637e-13	1.545430e+01	7.378967e-01	0.903961	0.053216	13
1000	100	3.882353e-14	1.120629e+01	1.750944e+00	0.750045	0.023976	12
			Triangle wave				
1000	300	2.764943e-10	6.319097e+00	9.239564e-02	0.964813	0.069409	13
1000	150	2.290984e-11	4.211512e+00	8.833424e-02	0.966517	0.031550	13
1000	50	1.364053e-11	2.986549e+00	1.121243e-01	0.959723	0.007290	11
			Sawtooth wave				
1000	300	1.037120e-11	1.217783e+01	4.255144e-01	0.826686	0.076458	13
1000	200	1.215180e-11	1.100427e+01	6.029744e-01	0.746856	0.053002	12
1000	100	3.162498e-12	6.582426e+00	8.374424e-01	0.614149	0.022410	12

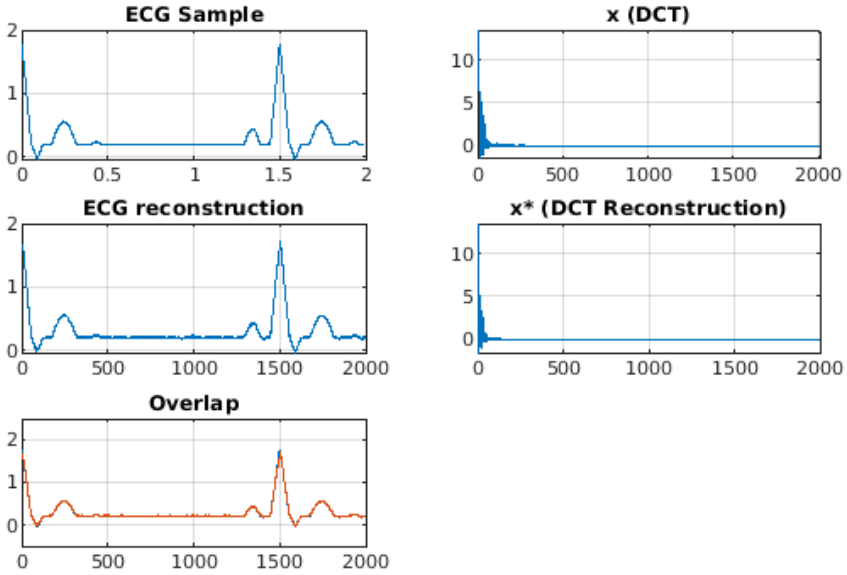


Figure A1. ECG signal sampled at 1kHz for 2 seconds having an average of 60 beats per minute. The overall pipeline is depicted here in a top-down sequence, the signal undergoes sampling and discrete cosine transform obtaining the original vector x. After solving the optimization system we retrieve x^* from which we performIDCT and

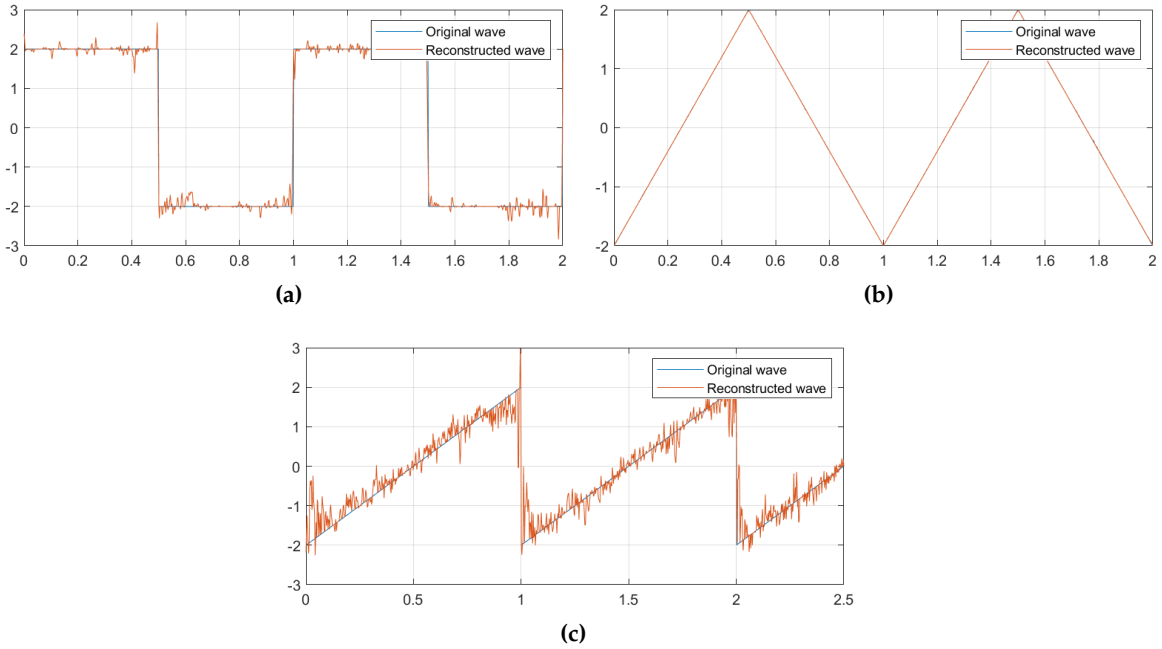


Figure A2. IPM method solutions overlapping of the original signal in blue $f(x)$ used for sampling and the reconstructed sum of cosines approximation $\hat{f}(x)$, (**a**) Square wave using n=1000 and m = 500, (**b**) Triangle wave using n=1000 and m = 300, (**c**) Sawtooth wave using n=1000 and m = 300.

References

1. Lustig, M.; Donoho, D.L.; Santos, J.M.; Pauly, J.M. Compressed Sensing MRI. *IEEE Signal Processing Magazine* **2008**, *25*, 72–82.
2. Moler, C. Magic" reconstruction: Compressive sensing. *Cleves Corner, Mathworks NewsNotes* **2010**, pp. 1–4.
3. Candes, E.J.; Romberg, J.; Tao, T. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory* **2006**, *52*, 489–509.
4. Li, H.L. An efficient method for solving linear goal programming problems. *Journal of Optimization Theory and Applications* **1996**, *90*, 465–469.
5. Bigi, G.; Frangioni, A.; Gallo, G.; Pallottino, S.; Scutellà, M. Appunti di Ricerca Operativa. 2007.
6. Nocedal, J.; Wright, S.J. *Numerical optimization*; Springer, 2006.