Exercise 2.1.a

Given $\mathcal{X} = \{X : Tr(X) \leq K, X \in \mathbb{C}^{p \times p}, X \succeq 0\}$, $\mathcal{X}$ is a convex set if, $\forall\ X, Y \in \mathcal{X}$ and $\lambda \in [0,1]$

then $Z = \lambda X + (1-\lambda) Y \in \mathcal{X}$

Since $X, Y \in \mathbb{C}^{p \times p}$ also $Z \in \mathbb{C}^{p \times p}$ because it's a linear combination of X, Y and ~~the~~ linear operations preserve dimensionality.

By definition of λ, $\lambda \geq 0$ and $(1-\lambda) \geq 0$, then $\lambda X \succeq 0$, $(1-\lambda) Y \succeq 0$

and, then, $Z \succeq 0$, since $X, Y \succeq 0$ by definition of $\mathcal{X}$.

Again, by definition of $\mathcal{X}$ $Tr(X) \leq K$ and $Tr(Y) \leq K$.

Let's suppose $Tr(X) = Tr(Y) = K$, then:

$$Tr(Z) = \lambda Tr(X) + (1-\lambda) Tr(Y) = \lambda K + (1-\lambda) K = K$$

Hence K is the maximum value $Tr(Z)$ can have, the $Tr(Z) \leq K$.

Then, $Z \in \mathcal{X}$ and $\mathcal{X}$ is a convex set.

Exercise 2.1.b

As done in Lecture 12 slide 27/51, we have that:

$$\min_{x \in \mathcal{X}} \{ f(x) : A_1(x) = b_1, A_2(x) = b_2, B(x) \in \mathcal{K} \}$$

$$\xleftrightarrow{\frac{1}{\beta} \to \infty} f(x) + \frac{1}{2\beta} \| A_1(x) - b_1 \|^2 + \frac{1}{2\beta} \| A_2(x) - b_2 \|^2 + \frac{1}{2\beta} dist^2(x, \mathcal{K})$$

$= F_\beta(x)$, with $F_\beta(x)$ being the penalized function with penalty parameter $\beta > 0$. Here $A_1(x) = X\mathbb{1}$, $A_2(x) = X^T\mathbb{1}$, $b_1 = 1$, $b_2 = 1$. Then, by applying the Danskin's theorem as mentioned in the hint, we rewrite $F_\beta(x)$ as follows:

$$F_\beta(x) = f(x) + \frac{1}{2\beta} \| A_1(x) - b_1 \|^2 + \frac{1}{2\beta} \| A_2(x) - b_2 \|^2 + \frac{1}{2\beta} \| K^* - x \|^2$$

where $K^* = proj_{\mathcal{K}}(x)$. Then:

$$\nabla F_\beta(x) = \nabla f(x) + \frac{1}{\beta}\left(A_1^T (A_1(x) - b_1)\right) + \frac{1}{\beta}\left(A_2^T (A_2(x) - b_2)\right) + \frac{1}{\beta}(X - K^*)$$

We can see that $\nabla F_\beta(x) = V_K / \beta$ in the algorithm.

Exercise 2.1.c

$$f(x) = \langle C, x \rangle \Rightarrow \nabla f(x) = \nabla \langle C, X \rangle = \nabla \, Tr(X^T C) = C$$

$\mathcal{K}$ operates element-wise, hence, given $Z \in \mathbb{R}^{p \times p}$ we define its elements with $Z_{i,j}$ with $i,j = 1, \dots, p$. $B(Z) \in \mathcal{K} \iff Z \geq 0$, hence the projection of each element $Z_{i,j}$ is $Z_{i,j}$ if $Z_{i,j} \geq 0$, $-Z_{i,j}$ otherwise.

In other words, $\text{proj}_{\mathcal{K}}(Z_{i,j}) = \underset{x_{i,j}}{\text{argmin}} \, |x_{i,j} - Z_{i,j}|$. By definition of $\mathcal{K}$ $x_{i,j}$ can't be negative, so if $Z_{i,j} < 0$ the solution is $X_{i,j} = 0$.

If $Z_{i,j} > 0$, the solution is $X_{i,j} = Z_{i,j}$.

Then, $\text{proj}_{\mathcal{K}}(Z_{i,j}) = \max(0, Z_{i,j})$, if applied element-wise to the matrix Z we obtain $\text{proj}_{\mathcal{K}}(Z)$.

Hence:

$$V_k = \beta_k \cdot C + A_1^T(A_1(X_k) - b_1) + A_2^T(A_2(X_k) - b_2) + X_k - \text{proj}_{\mathcal{K}}(X_k)$$

with $\text{proj}_{\mathcal{K}}(X_k)$ defined as previously explained.

Exercise 2.1.d

Using Moreau's decomposition we have that:

$$y^{k+1} = \text{prox}_{\sigma g^*}\left(y^k + \sigma A(\tilde{x}^{k+1})\right) = y^k + \sigma A(\tilde{x}^{k+1}) - \sigma \, \text{prox}_{\sigma^{-1} g}\left(\sigma^{-1} y^k + A(\tilde{x}^{k+1})\right)$$

Then $\text{prox}_{\delta_{\mathcal{K}}}(Z) = \text{proj}_{\mathcal{K}}(Z)$ as proved in problem 1.1.a and $g(z) = \delta_{\{b_1\}}(z_1) + \delta_{\{b_2\}}(z_2) + \delta_{\mathcal{K}}(z_3)$, so, given $y^k = [y_1^k, y_2^k, y_3^k]^T$ and $A = [A_1, A_2, I]^T$, we have that: $\text{prox}_{\sigma^{-1} g}(\sigma^{-1} y^k + A(\tilde{x}^{k+1})) =$

$$= \left[\text{prox}_{\sigma^{-1}\delta_{\{b_1\}}}(\sigma^{-1} y_1^k + A_1(\tilde{x}^{k+1})), \text{prox}_{\sigma^{-1}\delta_{\{b_2\}}}(\sigma^{-1} y_2^k + A_2(\tilde{x}^{k+1})), \text{prox}_{\sigma^{-1}\delta_{\mathcal{K}}}(\sigma^{-1} y_3^k + \tilde{x}^{k+1})\right]^T$$

$$= \left[b_1, b_2, \text{proj}_{\mathcal{K}}(\sigma^{-1} y_3^k + \tilde{x}^{k+1})\right]^T$$

Exercise 2.1.d : continuous

Hence:

$$y^{K+1} = y^K + \sigma\left(A(\tilde{x}^{K+1}) - \operatorname{prox}_{\sigma^{-1}g}\left(\sigma^{-1} y^K + A(\tilde{x}^{K+1})\right)\right) =$$

$$= \begin{bmatrix} y_1^K \\ y_2^K \\ y_3^K \end{bmatrix} + \sigma \begin{bmatrix} A_1(\tilde{x}^{K+1}) - b_1 \\ A_2(\tilde{x}^{K+1}) - b_2 \\ \tilde{x}^{K+1} - \operatorname{proj}_X\left(\sigma^{-1} y_3^K + \tilde{x}^{K+1}\right) \end{bmatrix}$$

Now, $A^T = [A_1^T, A_2^T, I]^T$, and with y^{K+1} from the expression above:

$$A^T y^{K+1} = A^T y^K + \sigma\left(A_1^T \cdot (A_1(\tilde{x}^{K+1}) - b_1) + A_2^T(A_2(\tilde{x}^{K+1}) - b_2) + \tilde{x}^{K+1} - \operatorname{proj}_X\left(\sigma^{-1} y_3^K + \tilde{x}^{K+1}\right)\right)$$

Exercise 2.1.e

By running HCGM and Vu-Condat to solve the k-means clustering problem, we obtained similar results. In fact, the initial k-means value before clustering was 150.9680, and after running the algorithms we obtained a k-means value of 28.7269 for both. Running Llyod's algorithm for ten times gave always different results, ranging from 28.7269 to 281.1941. The average result was a k-means value of 191.0464, which is a clear proof of why the Llyod's algorithm is not the best, since it mostly converges to local optimal points and the fact that the lowest value reached is the same as the one obtained by both HCGM and Vu-Condat gives us a proof that it is actually the global optimum.

The final objective value was 52.6948 for HCGM and 57.0374 for Vu-Condat. They're both below the provided optimal value provided of 57.05339187. This is because the solution X found by the algorithms does not necessarily respect all the constraints, since doing so is penalized but not forbidden by the implementation. Hence, as opposed to the optimal solution, X may contain some negative values that make the final objective value lower.

The misclassification rate for the two algorithms was 0.1250, but Vu-Condat is preferable as it converges faster, taking 303 seconds versus 569 seconds of HCGM (after just 1000 iterations, whereas HCGM needed 5000).

In figure 5 the plots are reported, whereas figures 6 and 7 show a visualization of the result on some samples.

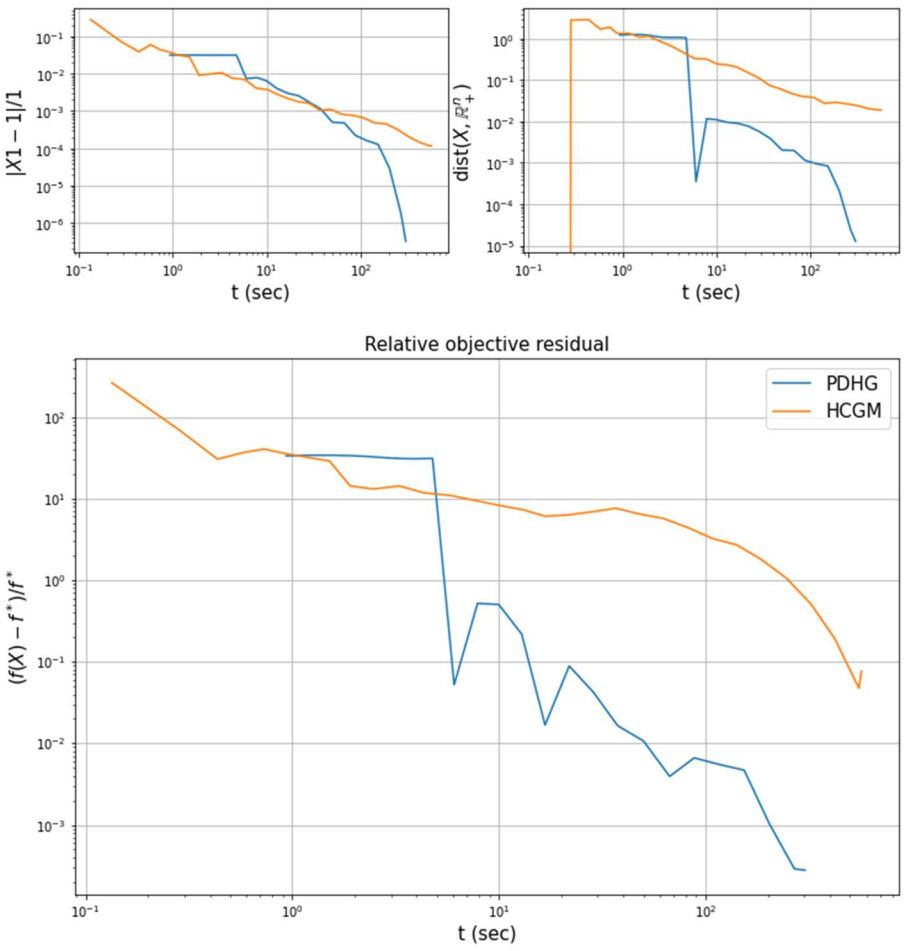

Figure 5: Convergence (top) and relative objective residual (bottom) for k-means clustering

Figure 6: Examples of results for HCGM

Figure 7: Examples of results for Vu-Condat

Exercise 3.a

In problem (3) from Part 2 the constraint $X \succeq 0$ had to be applied for each entry. In a matrix $X \in \mathbb{R}^{p \times p}$ we have p^2 entries, so the number of constraints was $O(p^2)$, with the other constraints being independent of p as they were applied to the matrix as a whole. The same goes for problem (7), with some constraints that are applied to the whole matrix and $B_{i,j,k}(X) \in \mathcal{K} = (-\infty, 0]$ which is a p-dependent constraint.

In fact, this constraint has to be applied to each and every distinct triplet, where the order of its elements counts.

From combinatorics we know that there are $p \cdot (p-1) \cdot (p-2)$ possible triplets, so the number of constraints is $O(p^3)$.

Exercise 3.b

As done in Lecture 12 Slide 27/51, we have that:

$$\min_{x \in \mathcal{X}} \left\{ f(x) : A(x) = \frac{p^2}{2}, B_{i,j,k}(X) \in \mathcal{K} \right\}$$

$$\Longleftrightarrow^{\mu \to \infty} f(x) + \frac{\mu}{2} \left\| A(x) - \frac{p^2}{2} \right\|^2 + \frac{\mu}{2} \operatorname{dist}^2(x, \mathcal{K}) = F_\mu(x), \text{ with}$$

$F_\mu(x)$ being the penalized function with penalty parameter $\mu > 0$.

Here $\mathcal{X} = \{X : \operatorname{Tr}(X) \leq p, X \in \mathbb{R}^{p \times p}, X \succeq 0\}$, $A(x) = p \cdot \operatorname{Tr}(x) - \operatorname{Tr}(\mathbf{1}_{p \times p} X)$ and $\mathcal{K} = (-\infty, 0]$

Exercise 3.c

From 3.a we have that, in addition to two constraints applied matrix-wise, there is also a number of constraints applied to each triplet of matrix' elements, which is dependent of the number of nodes p. Hence, for the graph with 25 nodes we have $25 \cdot 24 \cdot 23 = 13\,800$ constraints, for the one with 55 nodes we have $55 \cdot 54 \cdot 53 = 157\,410$ constraints and finally $102 \cdot 101 \cdot 100 = 1\,030\,200$ constraints for the 102 nodes graph.

Exercise 3.c: continue

Running the algorithm for the 25 nodes dataset we obtained the plots shown in figure 8 and a running time of 106.843 seconds, while running the 55 nodes one gave the plots in figure 9 and required 1189.366 seconds. Finally, the 102 nodes dataset took 7645 seconds and resulted in the plots in figure 10. From such results we can see that the number of constraints to be computed is a big bottleneck of this solution on big graphs, since doubling the number of nodes makes the algorithm almost 10 times slower.

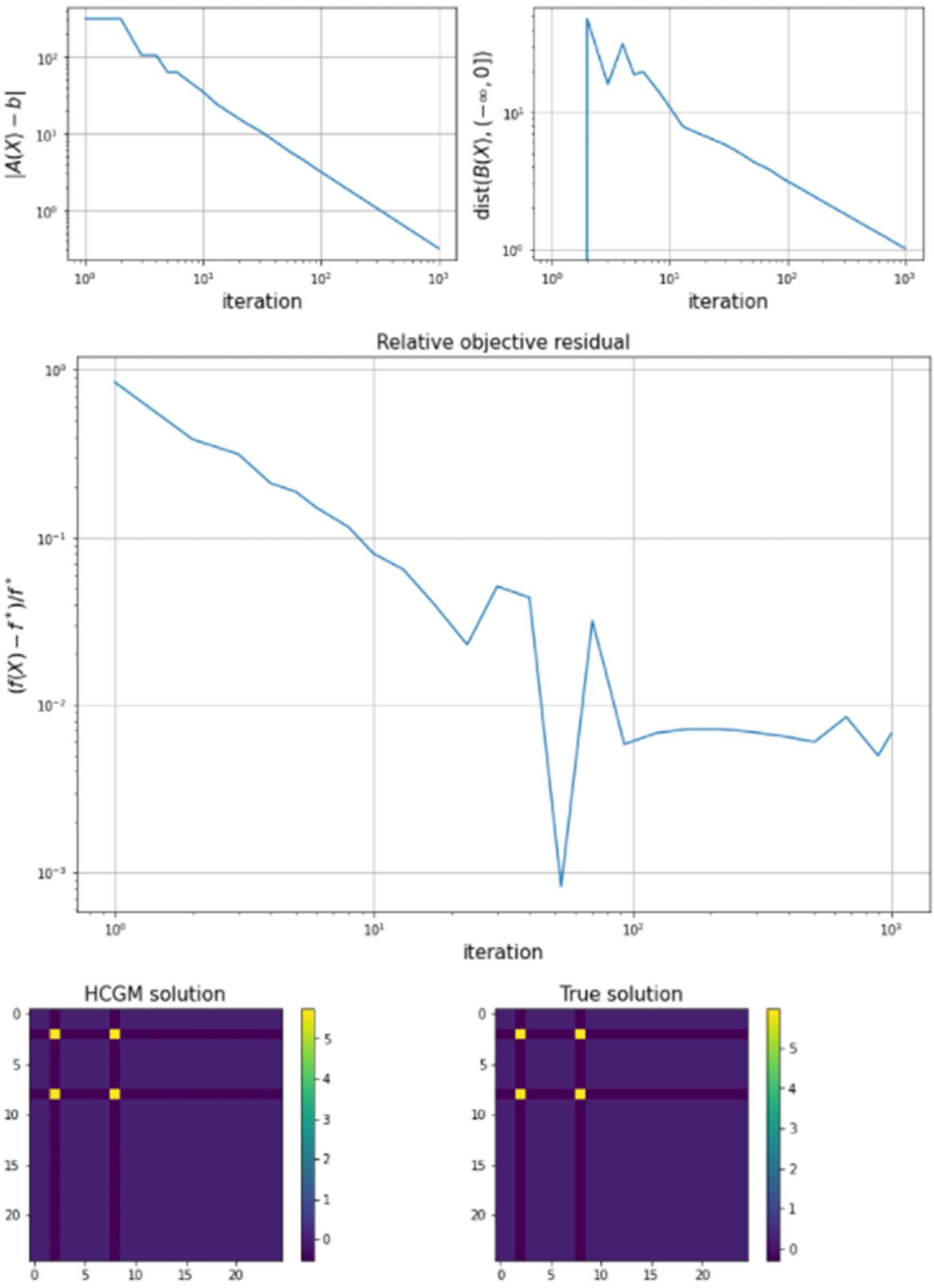

Figure 8 Results with the 25 nodes dataset

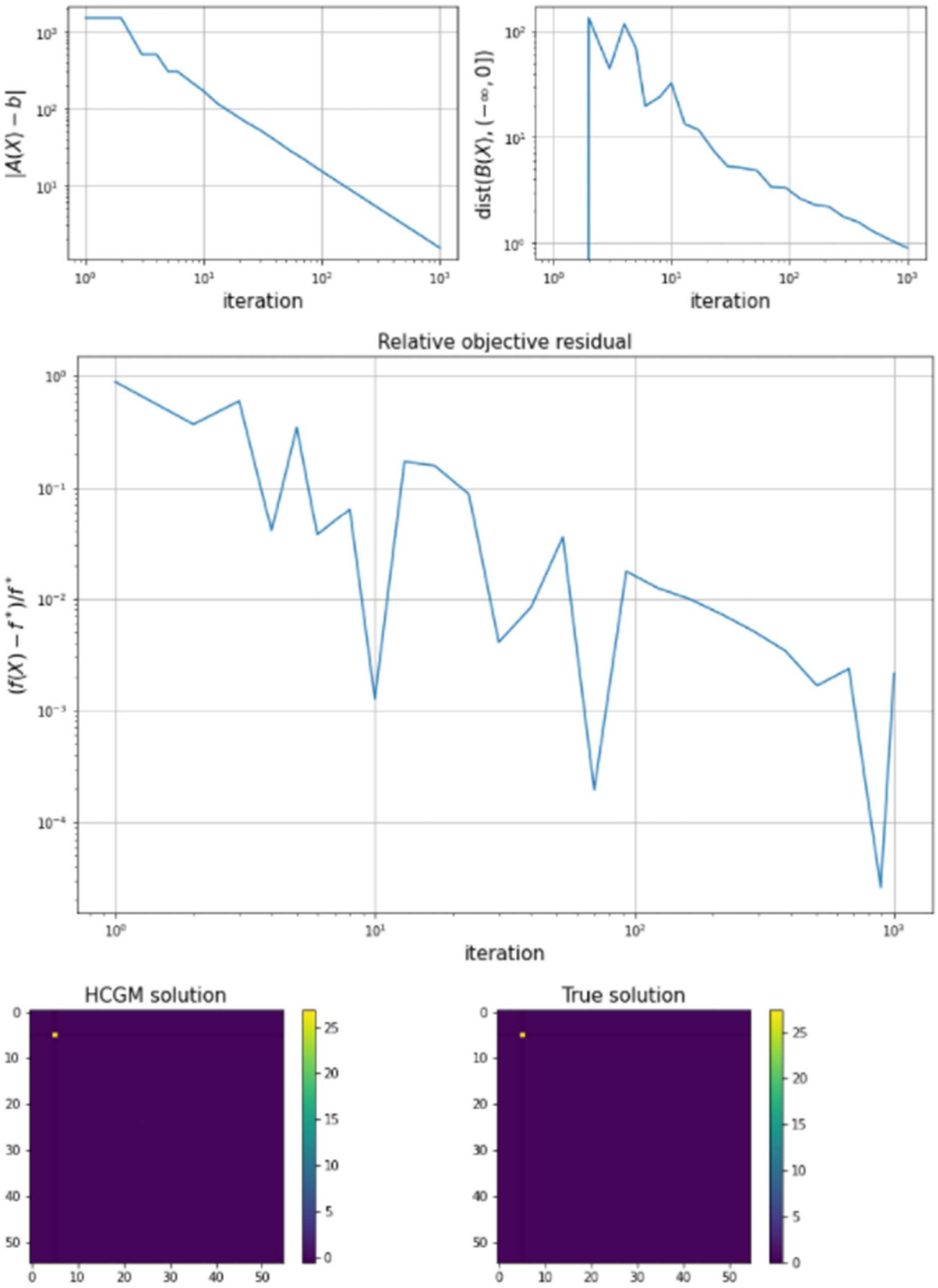

Figure 9 Results with the 55 nodes dataset

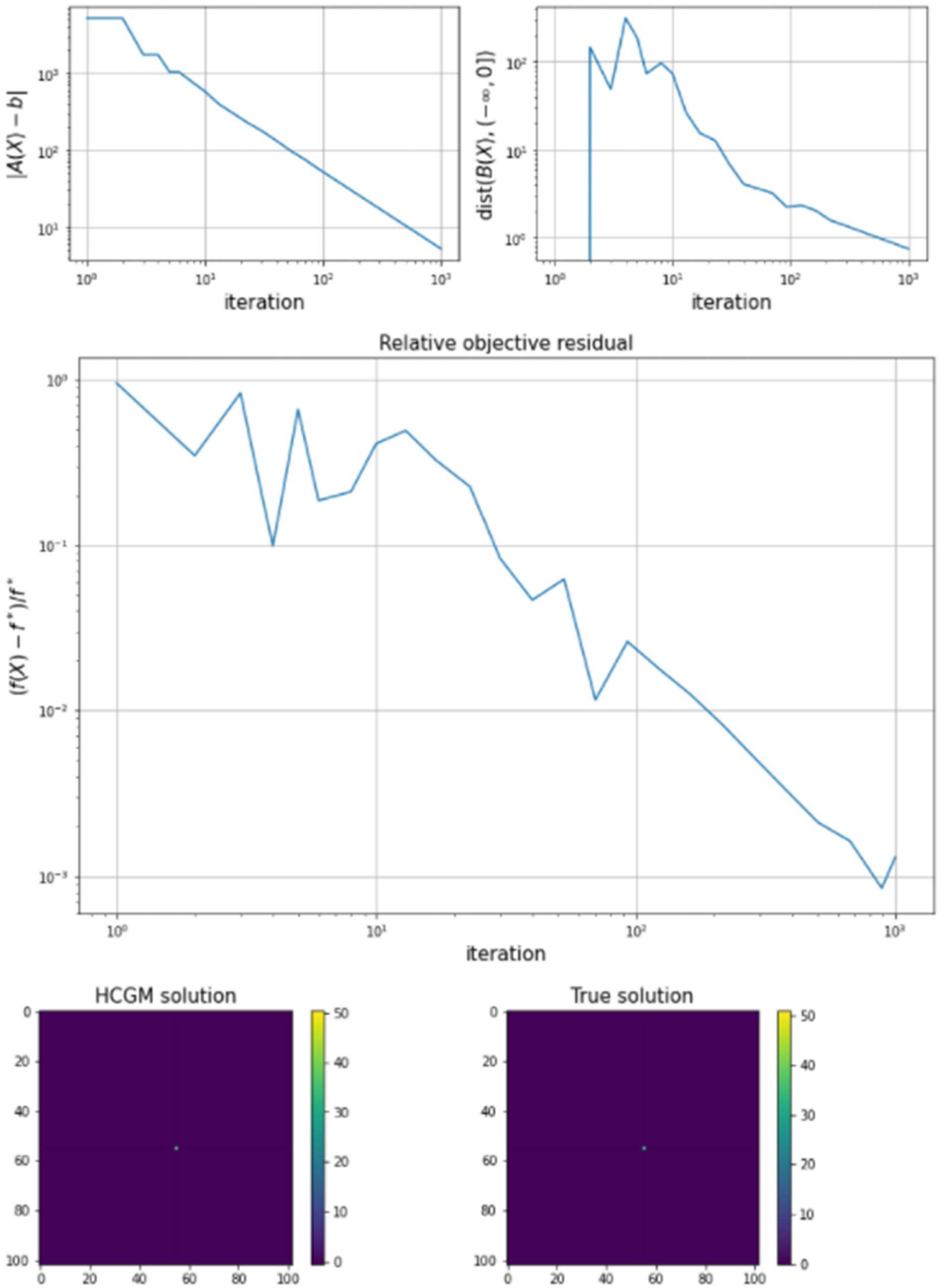

Figure 10 Results with the 102 nodes dataset