Università degli Studi di Padova

Optimization for Data Science Final Project

Minimum Enclosing Ball for Anomaly Detection on Biological Data

using Frank-Wolfe based algorithms

José Chacón, 2071875
Inês Jesus, 2073570
Alejandra Cruces, 2078632
Mario Tapia, 2081407

Department of Mathematics

September, 2023

Contents

1 Abstract

The Minimum Enclosing Ball (MEB) is a problem with a wide range of applications, of which anomaly detection is a prominent one. In this project, our goal is to understand and implement the MEB problem through three different variants of the Frank-Wolfe algorithm: the Pairwise Frank-Wolfe, the Blended Pairwise Conditional Gradient and an adaptation using Away-Steps.

Through the convergence analysis, we conclude that all of the algorithms have linear convergence when applied to the MEB problem.

To test and compare their performances, after exploring the theoretical results that appear in the literature, we use the implemented algorithms to find anomalies in the context of four different biological problems using different metrics, but focusing on the recall.

Finally, we will be able to observe that the third algorithm (from Yildirim (2008)) consistently outperforms the others across all four datasets in terms of computational time and number of iterations.

2 Introduction

The Minimum Enclosing Ball problem has a wide range of applications in clustering, data classification, machine-learning and facility location, among others. Due to its robustness, geometric interpretability and high-dimensionality, it is a valuable tool in the field of anomaly detection.

The dual MEB problem can be approached by the Frank Wolfe algorithm, its variants, and other closely related algorithms to Frank-Wolfe. Frank-Wolfe-type algorithms have been gaining popularity specially due to being projection-free and its sparsity, since at each iteration it expresses the solution as a convex combination of sparse atoms, guaranteeing the solution remains feasible and thus not needing to apply any projection strategies.

In this document, we will apply three of these types of algorithms proposed in the literature to determine the center and the radius of the hypersphere: the Pairwise Frank-Wolfe (Lacoste-Julien and Jaggi (2015) and Mitchell, Dem'yanov, and Malozemov (1974)), the Blended Pairwise Conditional Gradient (Tsuji, Tanaka, and Pokutta (2022)) and the $(1+\epsilon)$-approximation to the radius of the MEB algorithm (Yildirim (2008)), based on the Away-Steps Frank-Wolfe. As we will see on each of the convergence analysis, the three of them exhibit a linear convergence rate in the context of the MEB problem.

Moreover, we have chosen four different biological datasets: a Breast cancer dataset, a Breast cancer gene expression dataset, a Vertebral column pathology dataset and a Maternity risk dataset. It's for each of this data and context that we look to identify anomalies, which in the medical field and regarding patients, usually represents some sort of health problem.

Finally, we present and compare the obtained results regarding CPU time, core set size and number of iterations. The algorithm Yildirim purposes is by far the fastest to get a good approximation of the MEB and this difference is very obvious when working with large-scale datasets, such as the Breast Cancer gene expression one that contains 20,531 features. In practice, we get a similar behaviour for the PFW and BPCG algorithms.

3 Minimum Enclosing Ball Problem

Given $\mathcal{A} := \{a^1, \ldots, a^m\} \subset \mathbb{R}^n$, the Minimum Enclosing Ball (MEB) problem consists of finding the smallest Euclidean n-ball

$$\mathcal{B}_{c,\rho} := \{x \in \mathbb{R}^n : ||x - c||_2 \leq \rho\}$$

that contains every point in $\mathcal{A}$, considering c as the center, and $\rho \geq 0$ as the radius of the ball, respectively.

This problem can be formulated as

$$\begin{aligned} &\min_{c,\rho} \quad \rho \\ &\text{subject to} \\ &\qquad ||a^i - c|| \leq \rho, \quad i = 1, ..., m, \end{aligned}$$

from which, by applying the transformation $\gamma := \rho^2$, we obtain the equivalent optimization problem with smooth, convex quadratic constraints

$$\begin{aligned} (\mathcal{P}) \quad &\min_{c,\gamma} \quad \gamma \\ &\text{subject to} \\ &\qquad (a^i)^T a^i - 2(a^i)^T c + c^T c \leq \gamma, \\ &\qquad\qquad i = 1, ..., m. \end{aligned}$$

In high-dimensional spaces, transforming the MEB problem into its dual form can help reduce dimensionality and simplify the optimization process (see Appendix A for details on how to get the dual form of $(\mathcal{P})$ and proof of strong duality). So, we obtain

$$\begin{aligned} (\mathcal{D}1) \quad &\max_{\mu} \quad \phi(\mu) \\ &\text{subject to} \\ &\qquad \sum_{i=1}^{m} \mu_i = 1 \\ &\qquad \mu_i \geq 0, \quad i = 1, ..., m \end{aligned}$$

where

$$\phi(\mu) := \sum_{j=1}^{m} \mu_j (a^j)^T a^j - \left(\sum_{j=1}^{m} \mu_j a^j\right)^T \sum_{j=1}^{m} \mu_j a^j$$

and the vector μ contains the Lagrangian multipliers respective to problem $(\mathcal{P})$.

Since $\phi(\mu)$ is concave, we can reshape the concave maximization problem $(\mathcal{D}1)$ as an equivalent convex minimization problem.

Furthermore, considering the constraints of non-negativity ($\mu_i \geq 0, \quad i = 1, ..., m$) and sum-to-one ($\sum_{i=1}^{m} \mu_i = 1$), it is possible to infer that any feasible solution must reside in the unit simplex

$$\triangle^{m-1} = \left[\mu \in \mathbb{R}^m \,\middle|\, \sum_{i=1}^{m} \mu_i = 1 \quad \wedge \quad \mu_j \geq 0, \quad j = 1, ..., m\right].$$

Ultimately, we can express the dual formulation of the MEB ($\mathcal{D}1$) as a simplified convex version:

$$(\mathcal{D}2) \qquad \min_{\mu \in \triangle^{m-1}} -\phi(\mu)$$

4 Anomaly Detection

Anomaly detection is the task of identifying data points that deviate significantly from the normal behaviour of the data. Anomalies can indicate fraud, errors, faults, or other unusual events that may require further investigation.

One method for anomaly detection is to use the MEB approach, which consists of finding the smallest hypersphere that contains all the data points used for training, i.e., points that are known to not be anomalies *a priori*. From there, new data points that lie outside of the hypersphere obtained in the training stage are considered to be anomalies.

5 Algorithms

In this section, we will present three different approaches to the MEB problem, each one implemented based on variants of the classical Frank-Wolfe algorithm, and their respective convergence analysis studied in the literature.

We start with the Pairwise Frank-Wolfe (PFW) and the Blended Pairwise Conditional Gradients (BPCG) algorithms and then show how they were adapted to our problem in section 5.3.

The third algorithm is directly taken from Yildirim (2008) and is already an application of the Frank-Wolfe with away-steps designed to find a $(1+\epsilon)$-approximation to the MEB($\mathcal{A}$).

The algorithms consider general constrained convex optimization problems of the form

$$\min_{x \in P} f(x)$$

where P is a compact convex feasible region and f is convex and smooth with L-Lipschitz gradient over P.

5.1 Pairwise FW Algorithm

The Pairwise Frank-Wolfe (PFW) algorithm follows almost the same procedure as the Away-Steps Frank-Wolfe (Abadie, Wolfe, et al. (1970)), proposed to address the zig-zagging problem of the classical Frank-Wolfe algorithm. However, in the case of the PFW, the direction chosen in every step is based on a weight change from the away atom a_t to the FW atom w_t, while keeping all the other weights unchanged.

An issue with this algorithm is the possibility of occurrence of the so called *swap steps*, in which weight fully shifts from the away atom a_t to the FW atom w_t, i.e, the case where $\lambda_t = \lambda_{max}$ but $|S^{(t+1)}| = |S^{(t)}|$. However, under certain conditions it is possible to bound the number of swap steps that occur (see subsection 5.1.1).

Algorithm 1 Pairwise Frank-Wolfe algorithm: PFW($x^{(0)}$, $\mathcal{A}$, ϵ)

```
Require: point $x_0 \in \mathcal{A}$
  $S_0 \leftarrow \{x_0\}$
  for $t = 0$ to $T$ do
    $w_t \leftarrow LMO_A(\nabla f(x_t))$
    $d_t^{FW} = w_t - x_t$                                ▷ FW direction
    $a_t \leftarrow \text{argmax}_{v \in S_t} \langle \nabla f(x_t), v \rangle$
    $d_t^{A} = x_t - a_t$                                 ▷ Away direction
    $g_t^{FW} \leftarrow \langle -\nabla f(x_t), d_t^{FW} \rangle$
    if $g_t^{FW} \leq \epsilon$ then                      ▷ FW gap is small enough
      return $x^t$
    end if
    $d_t \leftarrow w_t - a_t$
    $\lambda_{max} \leftarrow \alpha_{a_t}$
    $\lambda_t \leftarrow \text{argmin}_{\lambda \in [0, \lambda_{max}]} f(x_t + \lambda d_t)$
    $x_{t+1} \leftarrow x_t + \lambda_t d_t$
    $\alpha_{a_t}^{(t+1)} = \alpha_{a_t}^{(t)} - \lambda_t$
    $\alpha_{w_t}^{(t+1)} = \alpha_{w_t}^{(t)} + \lambda_t$
    $S_{t+1} \leftarrow \{v \in A \text{ s.t. } \alpha_v^{t+1} > 0\}$
  end for
```

5.1.1 Convergence Analysis

As previously mentioned, Abadie et al. (1970) proposed to add the possibility to move away from an active atom in S_t to the Frank-Wolfe algorithm. This modification revealed to be sufficient to make the algorithm linearly convergent for strongly convex functions.

Theorem 1 from Lacoste-Julien and Jaggi (2015), assuming f has L-Lipschitz gradient and is μ-strongly convex over $M = \text{conv}(\mathcal{A})$, expresses that the suboptimality error $h_t := f(x^t) - f(x^*)$ of the iterates of the PFW decreases geometrically at each step that is not a drop step nor a swap step (i.e. when $\lambda_t < \lambda_{max}$, also called a 'good step'):

$$h_{t+1} \leq (1-\rho)h_t, \quad \text{where } \rho := \frac{\mu}{4L}\left(\frac{\delta}{M}\right)^2.$$

Here, $M = \text{diam}(\mathcal{M})$ and δ is a concept Lacoste-Julien and Jaggi introduce, called the *Pyramidal Width* of $\mathcal{A}$, which gives a lower bound on the angle formed between the negative gradient of f and the pairwise FW direction.

Moreover, if $k(t)$ is the number of 'good steps' up to iteration t, then we can only guarantee $k(t) \geq t/(3|A|!+1)$ because of the swap steps, which is not a satisfactory bound. Even so, Lacoste-Julien and Jaggi assure a global linear convergence rate for the PFW of the form

$$h_t \leq h_0 \exp(-\rho k(t)).$$

The dimension-dependence in the $k(t)$ bound is the reason the convergence proof of the PFW does not generalize to infinite-dimensional cases, which would be useful for kernel herding application, for example.

In our case, the objective function is not strongly convex. But Lacoste-Julien and Jaggi guarantee that the linear convergence of the PFW is still achieved for functions of the type $f(x) := g(Ax) + \langle b, x \rangle$ where g is continuously differentiable and strongly convex w.r.t. the Euclidean norm over the domain AP (keeping in mind that P is the feasible region of the problem). As we will see in section 5.3, problem ($\mathcal{D}2$) meets this requirement.

5.2 Blended Pairwise Conditional Gradients Algorithm

The second algorithm that we present proposes a modification of the Pairwise FW algorithm.

Taking ideas from the Blended Conditional Gradient Algorithm (Braun, Pokutta, Tu, and Wright (2018)), a blending criterion is added that favors local steps made over the convex hull of the current active vertex set. As a result, this algorithm offers a sparse solution. This is explained due to the only reason for new atoms being added to S_t is a sufficient decrease in the local pairwise gap.

This modification is sufficient to remove swap steps, an issue we have discussed that occurs with the PFW and this implies that the convergence rates of the BPCG are that which the PFW would achieve if swap steps would never occur. Another advantage is that it can be extended naturally to the infinite-dimension setting.

Here, the key idea is adding new vertices only when they are needed (when the local pairwise gap decreases sufficiently) and dropping vertices from the convex combination when a specific condition is met i.e. $\lambda_t > \Lambda_t^*$. When the latter case happens, it's called a *drop step*, whereas every other time a *descent step* is performed.

Algorithm 2 Blended Pairwise Conditional Gradients (BPCG)

```
Require: convex smooth function $f$, start vertex $x_0 \in V(P)$.
Ensure: points $x_1, \ldots, x_T$ in $P$
  $S_0 \leftarrow \{x_0\}$
  for $t = 0$ to $T-1$ do
      $a_t \leftarrow \text{argmax}_{v \in S_t} \langle \nabla f(x_t), v \rangle$          ▷ away vertex
      $s_t \leftarrow \text{argmin}_{v \in S_t} \langle \nabla f(x_t), v \rangle$          ▷ local FW
      $w_t \leftarrow \text{argmin}_{v \in V(P)} \langle \nabla f(x_t), v \rangle$        ▷ global FW

      if $\langle \nabla f(x_t), a_t - s_t \rangle \geq \langle \nabla f(x_t), x_t - w_t \rangle$ then
          $d_t = a_t - s_t$
          $\Lambda_t^* \leftarrow c[x_t](a_t)$
          $\lambda_t \leftarrow \text{argmin}_{\lambda \in [0, \Lambda_t^*]} f(x_t - \lambda d_t)$

          if $\lambda_t < \Lambda_t^*$ then
              $S_{t+1} \leftarrow S_t$                                   ▷ descent step
          else
              $S_{t+1} \leftarrow S_t \setminus \{a_t\}$                          ▷ drop step
          end if
      else
          $d_t = x_t - w_t$
          $\lambda_t \leftarrow \text{argmin}_{\lambda \in [0,1]} f(x_t - \lambda d_t)$
          $S_{t+1} \leftarrow S_t \cup \{w_t\}$ (or $S_{t+1} \leftarrow w_t$ if $\lambda_t = 1$)
                                                                  ▷ FW step
      end if
      $x_{t+1} \leftarrow x_t - \lambda_t d_t$
  end for
```

5.2.1 Convergence Analysis

The convergence rate on the BPCG is almost the same as in PFW but without swap steps. Because of this, it provides "state-of-the-art convergence guarantees for the strongly convex case" (Tsuji et al. (2022)). This modification also provides natural convergence rates for the infinite dimensional setting.

The authors define two theorems about the convergence rate for the function suboptimality.

Inequality (1) corresponds to the case when P is a convex feasible domain of diameter D, and f is convex and L-smooth. Inequalities (2) and (3) further require P to be a polytope with pyramidal width δ and f to be μ-strongly convex.

$$f(x_T) - f(x^*) \leq \frac{4LD^2}{T} \tag{1}$$

$$h_{t+1} \leq (1 - c_{f,P}) h_t \tag{2}$$

$$f(x_T) - f(x^*) \leq (f(x_0) - f(x^*)) \exp(-c_{f,P} T) \tag{3}$$

where $c_{f,P} := \frac{1}{2} \min\{\frac{1}{2}, \frac{\mu\delta^2}{4LD^2}\}$.

From Lemma 3.4 of Tsuji et al. (2022), we get $f(x_t) - f(x_{t+1}) \geq v$, where v is a value that changes depending of the type and size of step in the iteration. Let's declare it in the following way:

$$v = \begin{cases} \frac{\langle \nabla f(x_t), d_t \rangle^2}{2LD^2} & \text{, if t is a FW step with } \lambda_t^* < 1 \\ & \text{or a descent step} \\ \frac{1}{2}\langle \nabla f(x_t), d_t \rangle & \text{, if t is a FW step with } \lambda_t^* \geq 1 \end{cases}$$

This analysis so far does not take into account the possible drop steps that could happen during the iterations. So, in order to address this it's important to consider the number of FW steps, descent steps and drop steps, respectively identified as T_{FW}, T_{desc} and T_{drop}.

First, we note that $T = T_{\text{FW}} + T_{\text{desc}} + T_{\text{drop}}$ and because $|S_t| \geq 1$ we also know that necessarily $T_{\text{drop}} \leq T_{\text{FW}}$. Accordingly,

$$T = T_{\text{FW}} + T_{\text{desc}} + T_{\text{drop}} \leq 2T_{\text{FW}} + T_{\text{desc}} \leq 2(T_{\text{FW}} + T_{\text{desc}})$$

Considering the last inequality, the suboptimality conditions, for both settings of P and f, are possible to generalize for all the possible types of iterations. The detailed explanation is found in Tsuji et al. (2022).

5.3 MEB adaptation of PFW and BPCG

As seen in section 3, problem ($\mathcal{D}2$) is clearly a convex combination over a compact set and can be solved using Frank-Wolfe or its variants such as the Pairwise Conditional Gradient (Lacoste-Julien and Jaggi (2015) and Mitchell et al. (1974)) and the Blended Pairwise Conditional Gradient (Tsuji et al. (2022)).

Finding the step direction in algorithms 1 and 2, requires the computation of the gradient of $-\phi$, of which the components are given by

$$\nabla_i(-\phi(\mu)) = 2\left(a^i\right)^T \sum_{j=1}^{m} a^j \mu_j - (a^i)^T a^i.$$

In the case of Frank-Wolfe iterations on the PFW and BPCG algorithms, we compute w_t as:

$$w_t = \underset{w \in \triangle^{m-1}}{\text{argmin}}\, w^T \nabla\big(-\phi(\mu_t)\big) \tag{4}$$

From the Fundamental Theorem of Linear Programming, the minima of a linear function over a polytope are necessarily attained at the vertices. Therefore, the solution to (4) should be a vertex of $\triangle^{m-1}$.

Since the vertices of $\triangle^{m-1}$ correspond to the standard basis vectors $e_j \in \mathbb{R}^m$, $j = 1, ..., m$, problem 4 obtains the same result as

$$w_t = \underset{e_j \in \mathbb{R}^m}{\operatorname{argmin}} \, e_j^T \nabla\big(-\phi(\mu_t)\big) \quad .$$

So, if we take

$$i = i_t^w := \underset{j \in \{1,\dots,m\}}{\operatorname{argmin}} \Big(\nabla\big(-\phi(\mu_t)\big)\Big)_j \quad ,$$

w_t will simply be e_i.

The same simplification technique can be extended to the Away vertex, a_t, and Local Frank-Wolfe, s_t, presented in the pseudo-code from the PFW and BPCG algorithms with MEB adaptation, except in these cases taking into consideration only the active vertices from the previously defined active set S_t.

As such, in these cases, we get

$$i_t^a = \underset{\substack{j \in \{1,\dots,m\} \\ e_j \in S_t}}{\operatorname{argmax}} \Big(\nabla\big(-\phi(\mu_t)\big)\Big)_j$$

and

$$i_t^s = \underset{\substack{j \in \{1,\dots,m\} \\ e_j \in S_t}}{\operatorname{argmin}} \Big(\nabla\big(-\phi(\mu_t)\big)\Big)_j .$$

With these considerations, we present updated versions of the Pairwise Frank-Wolfe and Blended Pairwise Conditional Gradient now suitable for the MEB setting.

Algorithm 3 Pairwise Frank-Wolfe algorithm for the MEB problem: `PFW_MEB`($x^{(0)}$, $\mathcal{A}$, ϵ)

```
Require: point $x_0 \in \mathcal{A}$
  $S_0 \leftarrow \{x_0\}$
  for $t = 0$ to $T$ do
    $i_t^w \leftarrow \operatorname{argmin}_{j \in \{1,\dots m\}} (-\nabla\phi(\mu_t))_j$
    $d_t^{FW} = e_{\mu_t} - e_{i_t^w}$                                ▷ FW direction
    $i_t^a \leftarrow \operatorname{argmax}_{\substack{j \in \{1,\dots,m\} \\ e_j \in S_t}} (-\nabla\phi(\mu_t))_j$
    $d_t^A = e_{\mu_t} - e_{i_t^a}$                                   ▷ Away direction
    $g_t^{FW} \leftarrow \langle \nabla - \phi(\mu_t), d_t^{FW} \rangle$
    if $g_t^{FW} \le \epsilon$ then                      ▷ FW gap is small enough
      return $x^t$
    end if
    $d_t \leftarrow e_{i_t^w} - e_{i_t^a}$
    $\lambda_{max} \leftarrow \alpha_v^t$
    $\lambda_t \leftarrow \max\left(0, \min\left(\lambda_{max}, \frac{\langle -\nabla\phi(\mu_t), d_t \rangle}{L||d_t||_2^2}\right)\right)$
    $\mu_{t+1} \leftarrow \mu_t + \lambda_t d_t$
    $\alpha_{i_t^a}^{(t+1)} = \alpha_{i_t^a}^{(t)} - \lambda_t$
    $\alpha_{i_t^w}^{(t+1)} = \alpha_{i_t^w}^{(t)} + \lambda_t$
    $S_{t+1} \leftarrow \{e_{i_t^w} \in A \text{ s.t. } \alpha_v^{t+1} > 0\}$
  end for
```

As we mentioned in section 5.1.1, the linear convergence rates the PFW offers also generalize to our function, even though it doesn't exhibit strong convexity.

This is because our objective function

$$-\phi(\mu) = \left(\sum_{j=1}^{m} \mu_j a^j\right)^T \sum_{j=1}^{m} \mu_j a^j - \sum_{j=1}^{m} \mu_j (a^j)^T a^j$$

can be written using matrix representation like so:

$$-\phi(\mu) = \mu^t X^t X \mu - z^T \mu \quad ,$$

where $X = [a^1, ..., a^m]$ and $z_j = (a^j)^T a^j$.

We can now see that

$$\phi(\mu) = g(X\mu) + \langle -z, \mu \rangle$$

where $g(v) = v^T v$ is a strongly convex function.

Algorithm 4 Blended Pairwise Conditional Gradients (BPCG) for the MEB problem

```
Require: convex smooth function $f$, start vertex $\mu_0 \in \triangle^{m-1}$.
Ensure: points $\mu_1, \dots, \mu_T$ in $P$. Weights Initialization:
  $\alpha_t^v = 1$ for $v = \mu_0$, and 0 otherwise.
  $\mu_0 \leftarrow \frac{1}{m} * 1$                                ▷ Feasible initialization
  $S_0 \leftarrow \{\mu_0\}$
  $\lambda_t \leftarrow \frac{2}{t+2}$                                    ▷ Fixed step size
  for $t = 0$ to $T - 1$ do
    $i_t^a \leftarrow \operatorname{argmax}_{\substack{j \in \{1,\dots,m\} \\ e_j \in S_t}} (-\nabla\phi(\mu_t))_j$          ▷ away vertex
    $i_t^s \leftarrow \operatorname{argmin}_{\substack{j \in \{1,\dots,m\} \\ e_j \in S_t}} (-\nabla\phi(\mu_t))_j$           ▷ local FW
    $i_t^w \leftarrow \operatorname{argmin}_{j \in \{1,\dots m\}} (-\nabla\phi(\mu_t))_j$          ▷ global FW

    if $\langle \nabla - \phi(\mu_t), e_{i_t^a} - e_{i_t^s} \rangle \ge \langle \nabla - \phi(\mu_t), \mu_t - e_{i_t^w} \rangle$ then
      $d_t = e_{i_t^a} - e_{i_t^s}$
      $\Lambda_t^* \leftarrow c[\mu_t](e_{i_t^a})$
      $\lambda_t \leftarrow \max\left(0, \min\left(\Lambda_t^*, \frac{\langle \nabla\phi(\mu_t), d_t \rangle}{L||d_t||_2^2}\right)\right)$

      if $\lambda_t < \Lambda_t^*$ then
        $S_{t+1} \leftarrow S_t$                                ▷ descent step
        $\alpha^{e_{i_t^a}} = \alpha^{e_{i_{t-1}^a}} - \lambda_t$
        $\alpha^{e_{i_t^s}} = \alpha^{e_{i_{t-1}^s}} + \lambda_t$
      else
        $S_{t+1} \leftarrow S_t \setminus \{e_{i_t^a}\}$                          ▷ drop step
        $\alpha^{e_{i_t^a}} = 0$
        $\alpha^{e_{i_t^s}} = \alpha^{e_{i_{t-1}^s}} + \lambda_t$
      end if
    else
      $d_t = \mu_t - e_{i_t^w}$                                   ▷ FW step
      $\lambda_t \leftarrow \max\left(0, \min\left(1, \frac{\langle \nabla\phi(\mu_t), d_t \rangle}{L||d_t||_2^2}\right)\right)$
      $S_{t+1} \leftarrow S_t \cup \{w_t\}$ (or $S_{t+1} \leftarrow i_t^w$ if $\lambda_t = 1$)
      $\alpha_t = \alpha_{t-1}(1 - \lambda_t)$
      $\alpha^{e_{i_t^w}} = \alpha^{e_{i_t^w}} + \lambda_t$
    end if
    $\mu_{t+1} \leftarrow \mu_t - \lambda_t d_t$
  end for
```

For the line searches in Algorithms 3 and 4, we chose to use the short-step rule defined in Lemma 3.4 from Tsuji et al. (2022), for which their analysis still apply.

To determine the Lipschitz constant L, we start by calculating the Hessian matrix, H, of the objective function ϕ. The elements of this matrix are given by

$$h_{ij} = \frac{\partial^2 \left(-\phi(\mu)\right)}{\partial \mu_i \partial \mu_j} = 2 \left(a^j\right)^T a^i \quad .$$

and $L > 0$ is the largest eigenvalue of H.

We also test the two algorithms using a fixed *diminishing stepsize*

$$\lambda_t = \frac{2}{t+2}$$

and compare both approaches on the Numerical Experiments section.

5.4 (1+ϵ)-approximation to the MEB($\mathcal{A}$) algorithm

This algorithm is closely related to the Away-Steps variant of the Frank–Wolfe algorithm, with a proper initialization applied to the dual formulation of the MEB problem and the possibility of "dropping" points from the working core set at each iteration.

First, we introduce the notions of the $(1+\epsilon)$-approximation to the MEB($\mathcal{A}$) and core sets.

Given $\epsilon > 0$, a ball $\mathcal{B}_{c,\rho}$ is said to be a $(1+\epsilon)$-approximation to the MEB($\mathcal{A}$), $\mathcal{B}_{c_\mathcal{A},\rho_\mathcal{A}}$, if

$$\mathcal{A} \subset \mathcal{B}_{c,\rho} \quad \wedge \quad \rho \leq (1+\epsilon)\rho_\mathcal{A}.$$

A subset $\mathcal{X} \subseteq \mathcal{A}$ is said to be an ϵ-core set (or a core set) of $\mathcal{A}$ if

$$\rho_\mathcal{X} \leq \rho_\mathcal{A} \leq (1+\epsilon)\rho_\mathcal{X},$$

where $\mathcal{B}_{c_\mathcal{X},\rho_\mathcal{X}} :=$ MEB($\mathcal{X}$).

Core sets allow for a compact representation of the input set, and as such are of great importance when working with large-scale problems.

The way the following algorithm works is at each iteration it moves the center of the current ball either toward the furthest point of the core set or away from the closest, moving it by only a fraction of this distance. This happens in such a way that it constructs a sequence of balls with strictly increasing radius.

Algorithm 5 $(1+\epsilon)$-approximation to MEB($\mathcal{A}$)

```
Require: Input set of points $\mathcal{A} = \{a^1, \dots, a^m\} \subset \mathbb{R}^n, \epsilon > 0$.
  $\alpha \leftarrow \text{argmax}_{i=1,\dots,m} \|a^i - a^1\|^2$
  $\beta \leftarrow \text{argmax}_{i=1,\dots,m} \|a^i - a^\alpha\|^2$
  $u^0_\alpha \leftarrow 1/2,\ u^0_\beta \leftarrow 1/2$
  $\mathcal{X}_0 \leftarrow \{a^\alpha, a^\beta\}$
  $c^0 \leftarrow \sum_{i=1}^m u^0_i a^i$
  $\gamma^0 \leftarrow \Phi(u^0)$
  $\kappa \leftarrow \text{argmax}_{i=1,\dots,m} \|a^i - c^0\|^2$
  $\xi \leftarrow \text{argmin}_{i:a^i \in \mathcal{X}_0} \|a^i - c^0\|^2$
  $\delta^+_0 \leftarrow (\|a^\kappa - c^0\|^2/\gamma^0) - 1$
  $\delta^-_0 \leftarrow 1 - (\|a^\xi - c^0\|^2/\gamma^0)$
  $\delta_0 \leftarrow \max\{\delta^+_0, \delta^-_0\}$
  $k \leftarrow 0$
  while $\delta_k > (1+\epsilon)^2 - 1$ do
    if $\delta_k > \delta^-_k$ then
      $\lambda^k \leftarrow \delta_k/[2(1+\delta_k)]$
      $k \leftarrow k+1$
      $u^k \leftarrow (1-\lambda^{k-1})u^{k-1} + \lambda^{k-1}e^\kappa$
      $c^k \leftarrow (1-\lambda^{k-1})c^{k-1} + \lambda^{k-1}a^\kappa$
      $\mathcal{X}^k \leftarrow \mathcal{X}^{k-1} \cup \{a^k\}$
    else
      $\lambda^k \leftarrow \min\left\{\frac{\delta^-_k}{2(1-\delta^-_k)}, \frac{u^k_\xi}{1-u^k_\xi}\right\}$
      if $\lambda^k = u^k_\xi/(1-u^k_\xi)$ then
        $\mathcal{X}_{k+1} \leftarrow \mathcal{X}_k \setminus \{a^\xi\}$
      else
        $\mathcal{X}_{k+1} \leftarrow \mathcal{X}_k$
      end if
      $k \leftarrow k+1$
      $u^k \leftarrow (1+\lambda^{k-1})u^{k-1} - \lambda^{k-1}e^\xi$
      $c^k \leftarrow (1+\lambda^{k-1})c^{k-1} - \lambda^{k-1}a^\xi$
    end if
    $\gamma^0 \leftarrow \Phi(u^k)$
    $\kappa \leftarrow \text{argmax}_{i=1,\dots,m} \|a^i - c^0\|^2$
    $\xi \leftarrow \text{argmin}_{i:a^i \in \mathcal{X}_0} \|a^i - c^k\|^2$
    $\delta^+_k \leftarrow (\|a^\kappa - c^k\|^2/\gamma^k) - 1$
    $\delta^-_k \leftarrow 1 - (\|a^\xi - c^k\|^2/\gamma^k)$
    $\delta_k \leftarrow \max\{\delta^+_k, \delta^-_k\}$
  end while
  Output $c^k, \mathcal{X}_k, u^k, \sqrt{(1+\delta_k)\gamma^k}$
```

5.4.1 Convergence Analysis

According to Yildirim, the $(1+\epsilon)$-approximation to the MEB($\mathcal{A}$) algorithm converges in O($1/\epsilon$) iterations with an overall complexity bound of O(mn/ϵ) arithmetic operations and asymptotically exhibits linear convergence.

6 Numerical Experiments

In this section, we applied the three algorithms presented in the Section 5, PFW, BPFW, and $(1+\epsilon)$-approximation to the MEB($\mathcal{A}$), to four datasets from the biological field, and more specifically from the medical sector. To this end, we will present the metrics used to assess the capacity of our model for detecting anomalies, the preprocessing of the datasets, and finally the results of the numerical experiments.

6.1 Metrics

In the anomaly detection problem, we want to assess how well the model predicts the anomaly data in a new dataset. As we focus on the anomaly points, we use recall as the benchmark measure. Recall measures the fraction of the true anomalies detected by the model. Namely, we are interested in knowing how good our model is in detecting anomaly points over the total number of anomaly points.

Additionally, the precision and accuracy measure the fraction of detected anomalies that are true anomalies and the total points that are correctly classified, respectively.

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{7}$$

Where TP, FP, TN and FN represent the True Positives, False Positives, True Negatives, and False Negatives, respectively.

6.2 Datasets

A brief description of the four datasets is given.

1. Breast cancer dataset: This dataset contains 569 samples of women with breast tumors and 30 features of the tumor. The 29 real-valued features contain different measures of the tumor such as its radius, perimeter, smoothness, etc., and a dichotomous variable with the diagnosis, equal to 1 if the tumor is malignant and 0 if it is benign. The objective was to detect malignant tumors (anomalies) based on their characteristics.

2. Breast cancer gene expression dataset: This dataset is a collection of gene expressions of patients diagnosed with a certain type of tumor. It contains 801 instances and, due to the nature of the data, 20,531 features. For the purpose of this project, the focus is placed on breast cancer samples identified with the gen BRCA (Breast invasive carcinoma). With that on mind, the dataset has been adapted to an anomaly detection setting by declaring the breast cancer samples as normal and therefore other types of tumors as abnormal.

3. Vertebral column pathology dataset: This dataset contains 310 instances of patients with 6 biomechanical features derived from the shape and orientation of the pelvis and lumbar spine that are continuous values. Additionally, it has a categorical variable divided into patients without pathology and those with different vertebral column diseases. The goal was to detect if a patient has vertebral column pathology (anomaly) based on biomechanical indicators.

4. Maternity risk dataset: This dataset contains 1014 instances of pregnant women with 5 features of biomedical indicators (systolic and diastolic pressure, body temperature, among others), and a categorical variable if the woman had low, medium, or high risk of maternal mortality during pregnancy. We aim to identify pregnant women with high-risk levels during pregnancy (anomaly) based on medical characteristics.

Dataset	**Instances**	**Features**
Breast Cancer	569	30
Breast Cancer Gene	801	20,531
Vertebral Column	310	6
Maternity Risk	1014	6

Table 1: Datasets overview

6.3 Dataset Preprocessing

We divided the dataset into two sets: *training* and *test*, with a proportion of 70/30. The first was used to train the models and obtain the radius and center of the minimum enclosing ball containing only non-anomalous points. Then we used the radius and center of the computed MEB to evaluate the model's effectiveness in the test set. As mentioned before, we relied more heavily on recall to assess goodness of fit.

As seen on Table 1, we are working with datasets with different number of features. Therefore, additional preprocessing were made for specific datasets.

In the first dataset, we applied the StandardScaler function from *Scikit-Learn* library, that transforms the features of a dataset to have zero mean and unit variance. This is useful because it makes the data more comparable and reduces the effect of outliers and different scales. The data standardization is applied in both training and test set.

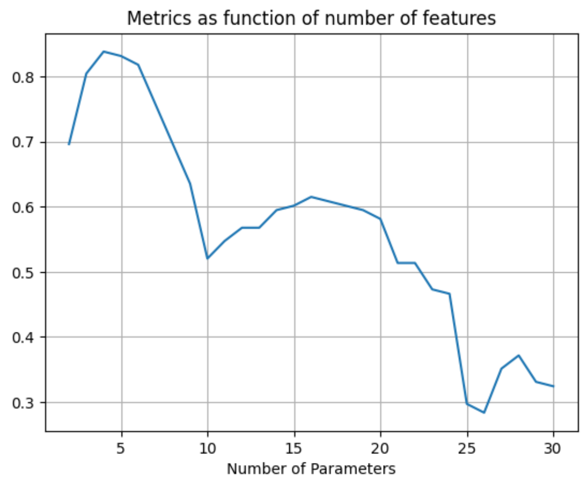

Figure 1: Optimal K search in Breast Cancer dataset based on the recall metric

Additionally, it was decided to reduce the dimensions of the data into a subset of K dimensions, for each dataset, in an effort to ensure the best value for the recall metric.

In order to settle on an optimal K value for our experiments, we performed a grid search on the K value using the BPCG algorithm. A result of this grid search is shown in Fig 1.

An exception was made for the Breast Cancer Gene Expression, given that its high dimensionality didn't have a considering diminishing effect on the recall metric, we considered this dataset didn't need an optimal K search. This data set will ultimately be used to show the difference in performance of the algorithms when applied to large-scaled problems.

Finally, with the optimal K chosen for each dataset as shown in Table 2, we were ready to start evaluating our algorithms performances.

Dataset	Features	Optimal K
Breast Cancer	30	4
Vertebral Column	6	2
Maternity Risk	6	3

Table 2: Optimal K search

6.4 Computational Results

First of all, in order to reduce the dimensionality of the datasets, we calculate the K-best features on the training set for each dataset using the *SelectKBest* function from *Scikit-Learn* library. The K used for each dataset is reported in Table 2.

We report convergence behaviour in the number of iterations and computational time for each of the algorithms and datasets. In all the experiments, we set a maximum number of iterations of $T = 100,000$ and a *tolerance* of 10^{-4} (which corresponds to the ϵ in the $(1+\epsilon)$- approximation to the MEB algorithm).

In the case of PFW and BPCG we use the same initial point and stopping condition. For Yildirim (2008) algorithm we used the proposed initial point X_0. Moreover, while the others algorithms check on the Frank Wolfe gap as a termination criterion, in the Yildirim (2008) algorithm the parameter δ is the stopping criteria. This parameter is an indicator of how much the radius should be expanded or reduced in order to be a valid minimum enclosing ball.

The numerical results are summarized on tables 3 and 4, where the second table uses diminishing step size for the computations.

	Time (ms)			**Iterations**		
Dataset	**PFW**	**BPCG**	**MEB(A)**	**PFW**	**BPCG**	**MEB(A)**
1	305.42	637.14	53.59	1,484	2,998	44
2	253,589.34	255,090.94	2,659.64	100,000	100,000	44
3	1,886.47	4,911.60	48.71	20,411	52,194	105
4	44,240.58	42,329.92	65.24	100,000	100,000	28

Table 3: Computational results with short step rule for PFW and BPCG

	Time (ms)		**Iterations**	
Dataset	**PFW**	**BPCG**	**PFW**	**BPCG**
1	65.40	31.50	396	163
2	332,794.27	334,992.27	100,000	100,000
3	18.76	155.47	158	2,155
4	62.19	70.12	168	292

Table 4: Computational results with diminishing step size

From table 3, we first notice that no matter the dataset, the MEB-approximation algorithm has better computational results, regarding both the time and number of iterations. We can see this difference more precisely in the experiments on the second dataset, where there's a reduction of around 99% in terms of iterations and computing time compared to the other algorithms. The other two algorithms took over six minutes to reach the maximum number of iterations allowed. This finding shows that it does have superior convergence rates when working with large-scale problems. One possible explanation is the absence of the gradient calculation in each iteration, which represents a costly operation. Instead of this, what the Yildirim (2008) algorithm computes in each iteration is the farthest point from the center and also its closest coreset point. Additionally, the initialization of the α and β as coreset points seems to give a good approximation of the optimal diameter from the very beginning.

This faster convergence of the third algorithm also is observed when compared to the computational results obtained with the other algorithms when using a diminishing step size, as shown in Table 4. Using this table, we aim to compare the performance of the algorithms using both step size strategies. We can see that the diminishing step size provides better computational results in general, although in the case of the second dataset exhibits an increase of 24% in the computing time when compared to the other step size strategy. This result indicates that using a Lipschitz constant dependent step size has a beneficial advantage when applied to large-scale instances.

The figures 2 to 5 complement what's been explained before, focusing on the behaviour of the algorithms over time/number of iterations, with respect to each of their convergence measure (FW gap for PFW and BPCG; δ for the MEB-approximation). In the case of the BPCG and PFW algorithms we only present the results for the short-step variant, the one for which the convergence analysis applies.

We want to highlight that the scales used are different for each plot, due to the difference in the magnitude of the values reached in each case.

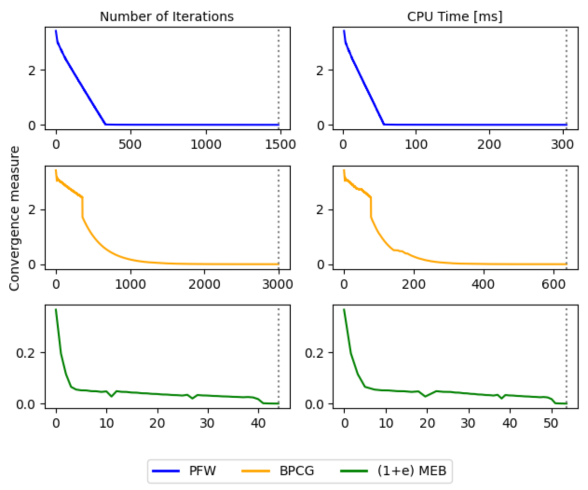

Figure 2: Convergence measure for Breast cancer dataset

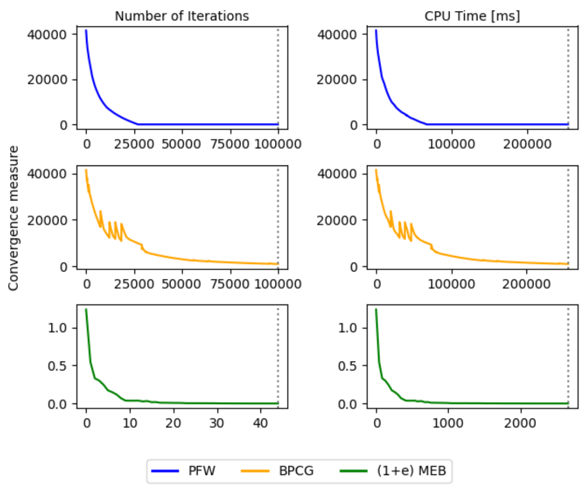

Figure 3: Convergence measure for Breast cancer gene expression dataset

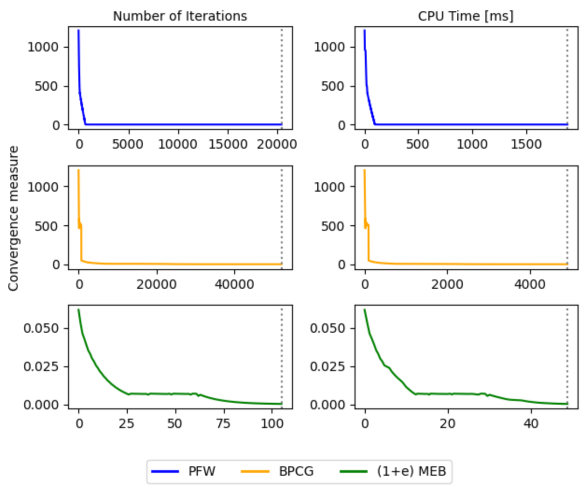

Figure 4: Convergence measure for Vertebral column pathology dataset

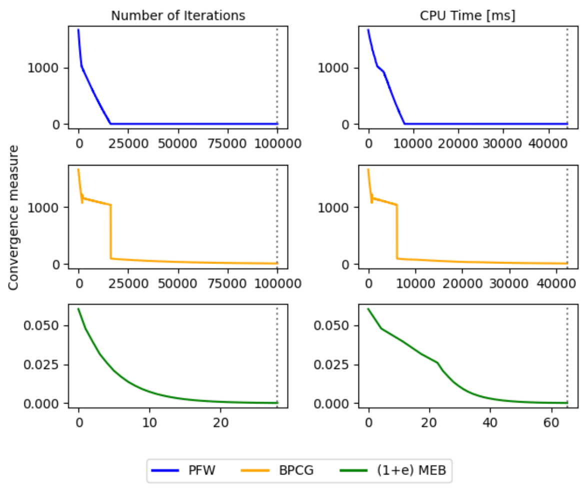

Figure 5: Convergence measure for Maternity risk dataset

For the convergence plots of these algorithms with the diminishing step size, they are to be found in appendix B.

	Recall Training			Recall Test		
Dataset	**PFW**	**BPCG**	**MEB(A)**	**PFW**	**BPCG**	**MEB(A)**
1	0.84	0.84	0.84	0.84	0.84	0.84
2	0.84	0.85	0.84	0.85	0.86	0.85
3	0.59	0.59	0.59	0.51	0.51	0.51
4	0.43	0.44	0.43	0.48	0.50	0.48

Table 5: Recall results

	Recall Training			Recall Test		
Dataset	**PFW**	**BPCG**	**MEB(A)**	**PFW**	**BPCG**	**MEB(A)**
1	0.84	0.84	0.84	0.84	0.84	0.84
2	0.84	0.85	0.84	0.85	0.85	0.85
3	0.59	0.59	0.59	0.51	0.51	0.51
4	0.43	0.44	0.43	0.48	0.48	0.48

Table 6: Recall results with diminishing step size

From table 5, we understand the general behaviour of the different algorithms for correctly identifying the anomalies. We emphasize that the recall is the rate of anomalous points that were correctly identified among all of the anomalous points in the dataset. As expected, all of the algorithms present very similar values of recall for each of the datasets and end up performing better for the first two datasets. This might be because these are more suitable for the MEB problem.

Additionally, the results regarding the computed radius and the active set at the moment of the last iteration are presented in table 7. It's clearly seen that the three algorithms arrive to a similar solution for the MEB problem, regarding these two variables. Given that in the results for the second dataset we study the behaviour of these algorithms when $m \gg n$, they indicate that as the dimensionality of the instances increases, so does the number of elements in the active set/core set.

	Radius			Active set/Core set size		
Dataset	**PFW**	**BPCG**	**MEB(A)**	**PFW**	**BPCG**	**MEB(A)**
1	1.9478	1.9478	1.9480	3	3	3
2	214.0098	213.9576	214.027	12	12	12
3	28.1097	28.1097	28.1125	3	3	3
4	39.4207	39.42042	39.4241	3	3	3

Table 7: Computational results

In Appendix B, we further document the improvement in the computed size of the radius through the iterations and computational time for both stepsize strategies.

7 Conclusion

In this work, we presented the MEB problem approach to the anomaly detection task by means of three different algorithms related to the classical Frank-Wolfe. The three algorithms allow drop steps, representing a significant improvement of the Frank-Wolfe algorithm, allowing sparser solutions. On the other hand, for the three algorithms applied to the MEB-problem we obtain a linear convergence rate. Moreover, the three algorithms keep throughout the iterations and ultimately return a sparse solution, which is a very good quality and helps making the computation times low.

The MEB method for anomaly detection was shown to be feasible by the numerical experiments performed. The results were satisfactory, especially for datasets 1 and 2, which achieved recalls around 85%. Datasets 3 and 4 had lower recalls around 50% for the test sets, which might be affected by how the instances are distributed in space and how close they are to a hypersphere shape.

The $(1+\epsilon)$-approximation to the MEB algorithm outperformed the other methods on the datasets, showing faster convergence and lower computational cost. This is specially true when working with large-scale datasets, case that a Yildirim highlights as being an advantage to use this algorithm.

References

[1] Abadie, J., Wolfe, P., et al. (1970). Integer and nonlinear programming.

[2] Braun, G., Pokutta, S., Tu, D., & Wright, S. (2018). *Blended conditional gradients: the unconditioning of conditional gradients.*

[3] Lacoste-Julien, S., & Jaggi, M. (2015). On the Global Linear Convergence of Frank-Wolfe Optimization Variants. In *Proceedings of the 28th international conference on neural information processing systems - volume 1* (p. 496–504). Cambridge, MA, USA: MIT Press.

[4] Mitchell, B., Dem'yanov, V. F., & Malozemov, V. (1974). Finding the point of a polyhedron closest to the origin. *SIAM Journal on Control*, *12*(1), 19–26.

[5] Tsuji, K., Tanaka, K., & Pokutta, S. (2022). Pairwise Conditional Gradients without Swap Steps and Sparser Kernel Herding. In *International conference on machine learning, ICML 2022, 17-23 july 2022, baltimore, maryland, USA* (Vol. 162, pp. 21864–21883). PMLR. Retrieved from `https://proceedings.mlr.press/v162/tsuji22a.html`

[6] Yildirim, E. A. (2008). Two algorithms for the minimum enclosing ball problem. *SIAM Journal on Optimization*, *19*(3), 1368-1391. Retrieved from `https://doi.org/10.1137/070690419`

A Lagrangian Dual of the MEB Problem

Attending to the Problem (3) described in Section 3,

$$(\mathcal{P}) \quad \min_{c,\gamma} \quad \gamma \quad \text{subject to} \quad (a^i)^T a^i - 2(a^i)^T c + c^T c \leq \gamma, \quad i = 1, ..., m,$$

our goal is to establish its dual form.

First, considering μ the vector where the j-th element corresponds to the Lagrange multiplier for the j-th constraint, we obtain the Lagrangian function as follows:

$$\begin{aligned}\mathcal{L}(c, \gamma, \mu) &= \gamma + \sum_{j=1}^{m} \mu_j \left((a^j)^T a^j - 2(a^j)^T c + c^T c - \gamma\right) \\ &= \gamma + \sum_{j=1}^{m} \mu_j (a^j)^T a^j - 2c^T \sum_{j=1}^{m} \mu_j a^j + c^T c \sum_{j=1}^{m} \mu_j - \gamma \sum_{j=1}^{m} \mu_j\end{aligned} \tag{8}$$

If c^* and γ^* feasibly solve the primal problem $(\mathcal{P})$, then there exists $\mu^* \in \mathbb{R}^m$ such that the Karush-Kuhn-Tucker stationarity condition is met, that is,

$$\frac{\partial \mathcal{L}(c^*, \gamma^*, \mu^*)}{\partial c} = 0 \quad \wedge \quad \frac{\partial \mathcal{L}(c^*, \gamma^*, \mu^*)}{\partial \gamma} = 0,$$

which gives us

$$c^* = \sum_{j=1}^{m} \mu_j^* a^j \quad \wedge \quad \sum_{j=1}^{m} \mu_j^* = 1. \tag{9}$$

As such, using the values obtained in equation (9) on equation (8), we get

$$\begin{aligned}\mathcal{L}(c^*, \gamma^*, \mu^*) &= \gamma^* + \sum_{j=1}^{m} \mu_j^* (a^j)^T a^j - 2\,(c^*)^T \sum_{j=1}^{m} \mu_j^* a^j + (c^*)^T c^* \sum_{j=1}^{m} \mu_j^* - \gamma^* \sum_{j=1}^{m} \mu_j^* \\ &= \gamma^* + \sum_{j=1}^{m} \mu_j^* (a^j)^T a^j - 2 \left(\sum_{j=1}^{m} \mu_j^* a^j\right)^T \sum_{j=1}^{m} \mu_j^* a^j + \left(\sum_{j=1}^{m} \mu_j^* a^j\right)^T \sum_{j=1}^{m} \mu_j^* a^j - \gamma^* \\ &= \sum_{j=1}^{m} \mu_j^* (a^j)^T a^j - \left(\sum_{j=1}^{m} \mu_j^* a^j\right)^T \sum_{j=1}^{m} \mu_j^* a^j \\ &=: \phi\left(\mu^*\right)\end{aligned} \tag{10}$$

Finally, we reach our dual formulation of the problem, which, by putting together equations (9) and (10) and the dual feasibility optimality condition, results in Problem $(\mathcal{D})$.

$$(\mathcal{D}) \quad \max_{\mu} \quad \phi(\mu) \quad \text{subject to} \quad \sum_{j=1}^{m} \mu_j = 1, \quad \mu_i \geq 0, \quad i = 1, ..., m$$

From solving this problem and getting μ^*, we already know from Equation (10) that we can obtain the center of the MEB, c^*. Now, if we also consider the complementary slackness Karush-Kuhn-Tucker condition, we will have

$$(\mu^*)^T \left((a^j)^T a^j - 2(a^j)^T c^* + (c^*)^T c^* - \gamma^*\right) = 0,$$

from which we get

$$\gamma^* = \phi(\mu^*).$$

This implies that strong duality holds between $(\mathcal{P})$ and $(\mathcal{D})$ and that μ^* is an optimal solution of $(\mathcal{D})$.

Finally, from γ^*, we obtain the optimal radius of the MEB $\rho^* = \sqrt{\phi(\mu^*)}$.

B Plots

B.1 Radius size using short step rule step size

As we performed the experiments for computing the Minimum Enclosing Ball, one of the most important elements returned by the algorithms is the radius.

The figures 6 to 9 shows that the radius increases progressively up to a certain iteration not that far from the start. From then, the updates start being very small until the algorithms stop. A special case is the one seen on the results of the approximation to the MEB proposed by Yildirim, where the algorithm is capable to obtain a better computation of the radius size in the first iteration.

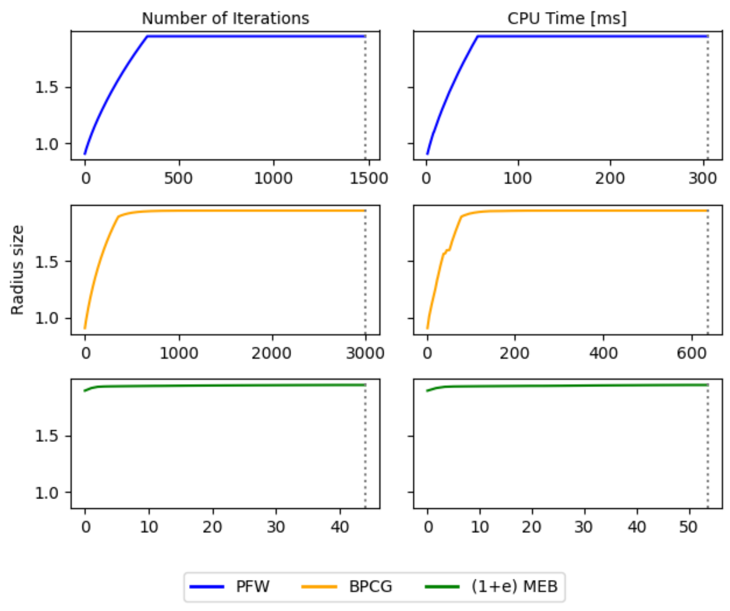

Figure 6: Radius size for Breast cancer dataset

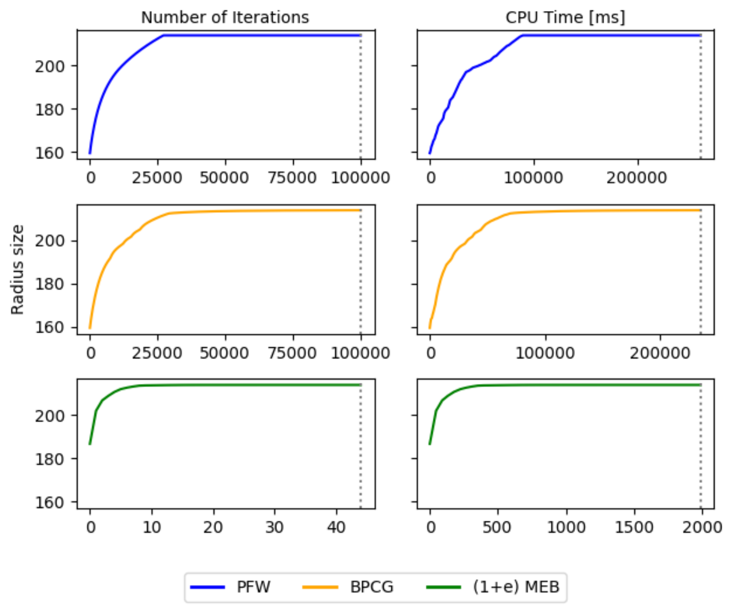

Figure 7: Radius size for Breast cancer gene expression dataset

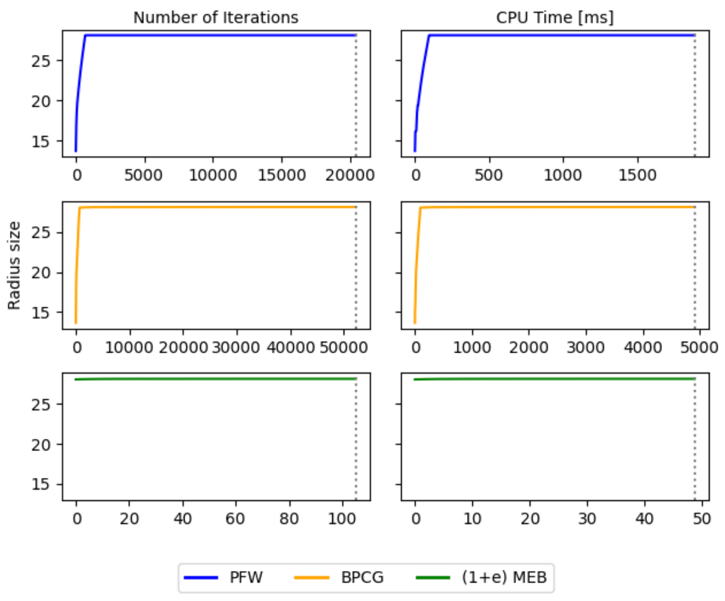

Figure 8: Radius size for Vertebral column dataset

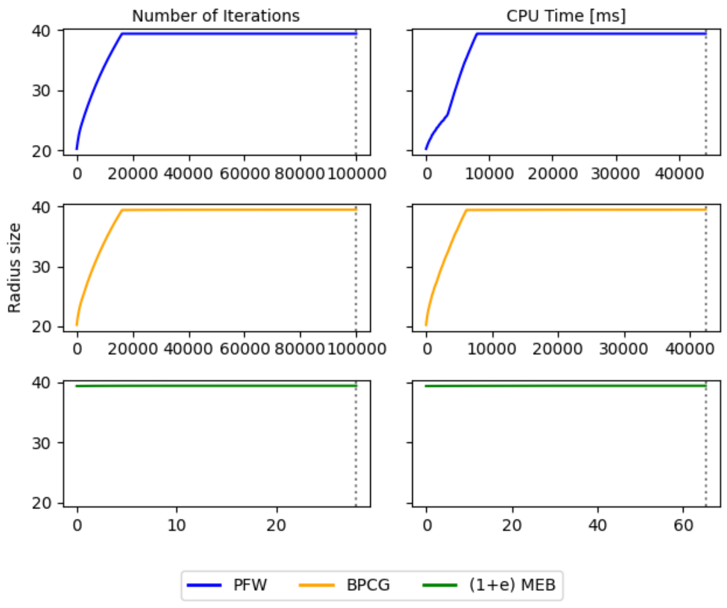

Figure 9: Radius size for Maternity risk dataset

B.2 Radius size using diminishing step size

The figures 10 to 13 exhibit similar increasing rates as the experiments ran by using the short step rule strategy

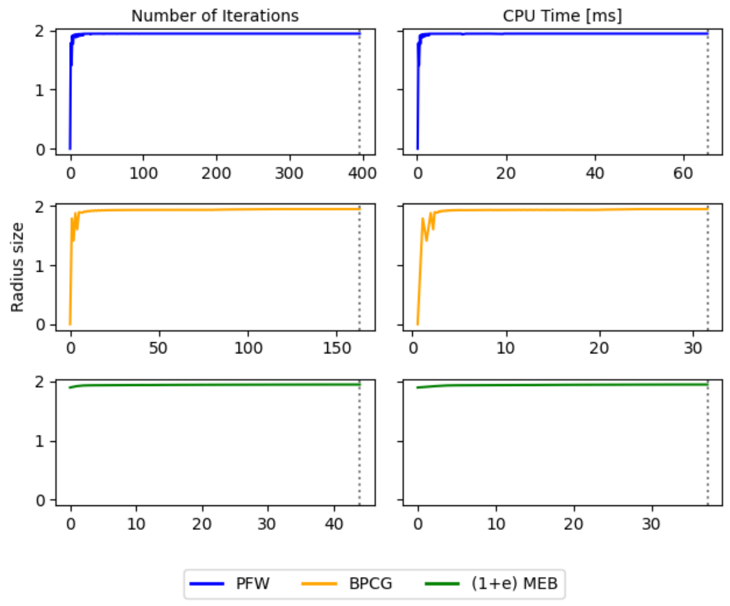

Figure 10: Radius size for Breast cancer dataset

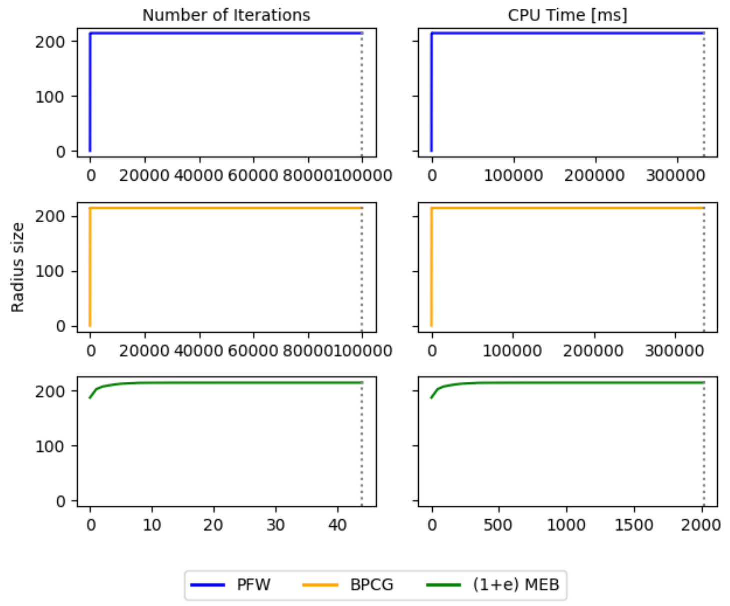

Figure 11: Radius size for Breast cancer gene expression dataset

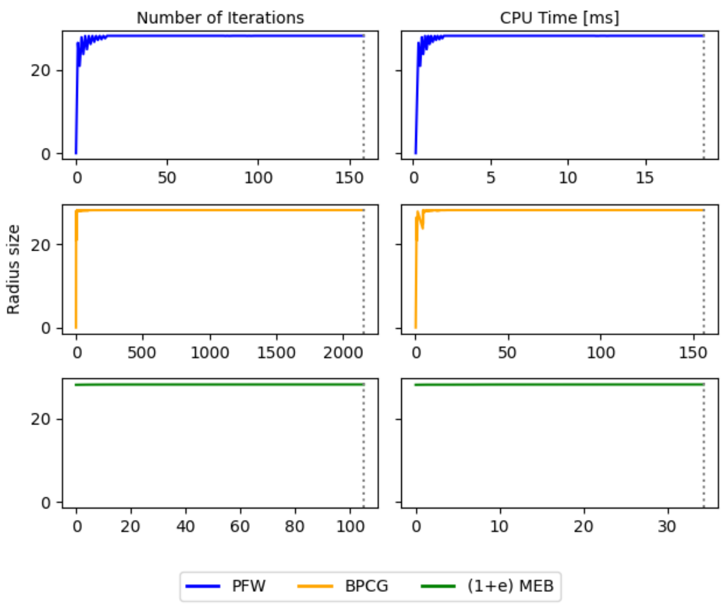

Figure 12: Radius size for Vertebral column dataset

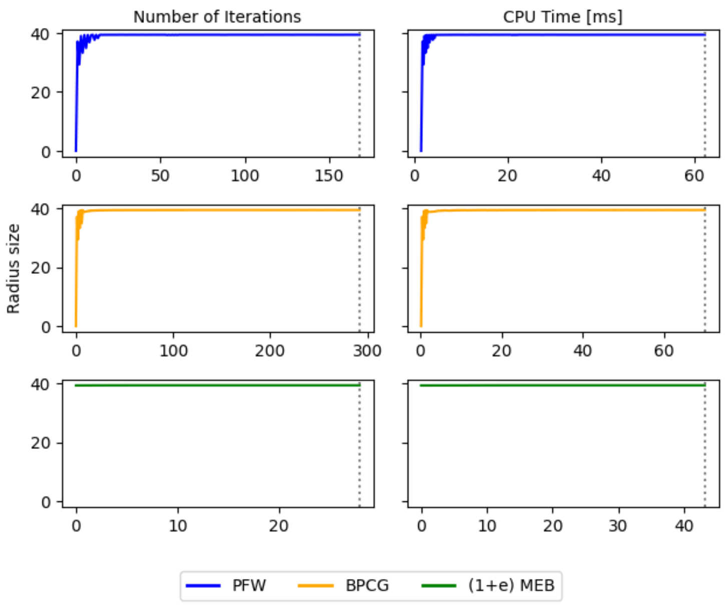

Figure 13: Radius size for Maternity risk dataset

B.3 Convergence measure using diminishing step size

This section expands on the convergence measure results found in our experiments,as shown in table 3. These plots indicates that when using this step size strategy, it takes less time to find a nice approximation to the MEB when dealing with smaller to medium sized datasets. Although, without the theoretical guarantees that provide the short step rule.

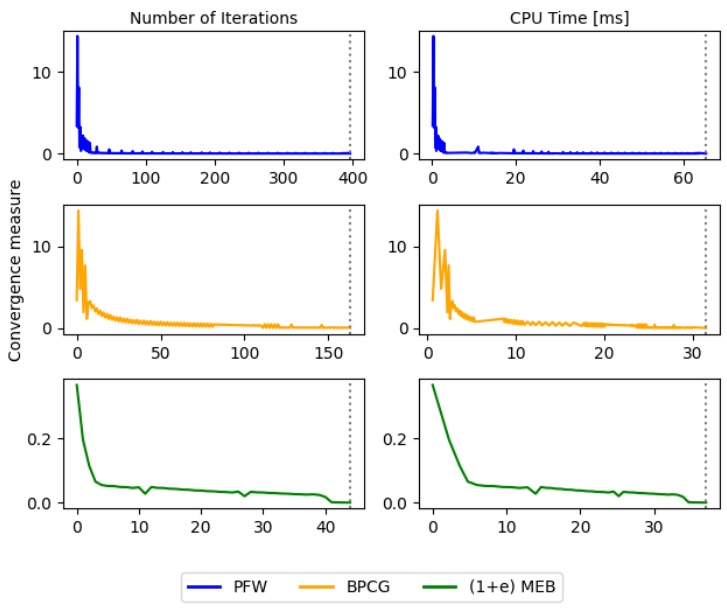

Figure 14: Convergence measure for Breast cancer dataset

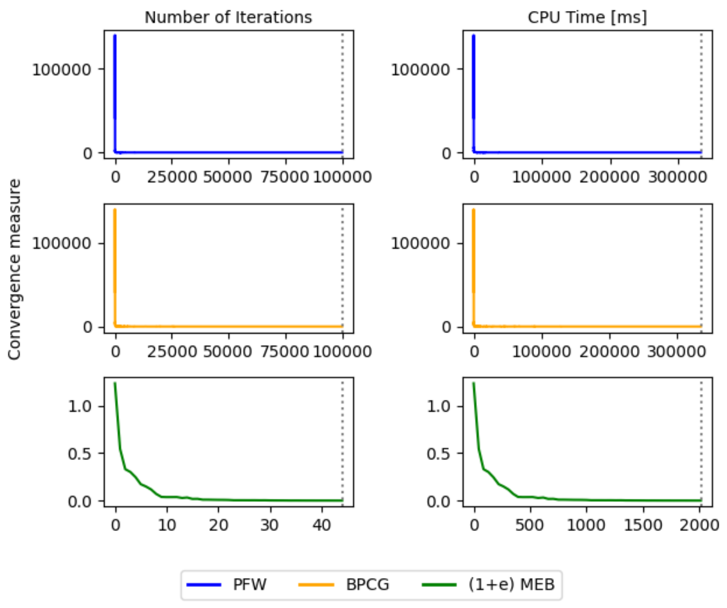

Figure 15: Convergence measure for Breast cancer gene expression dataset

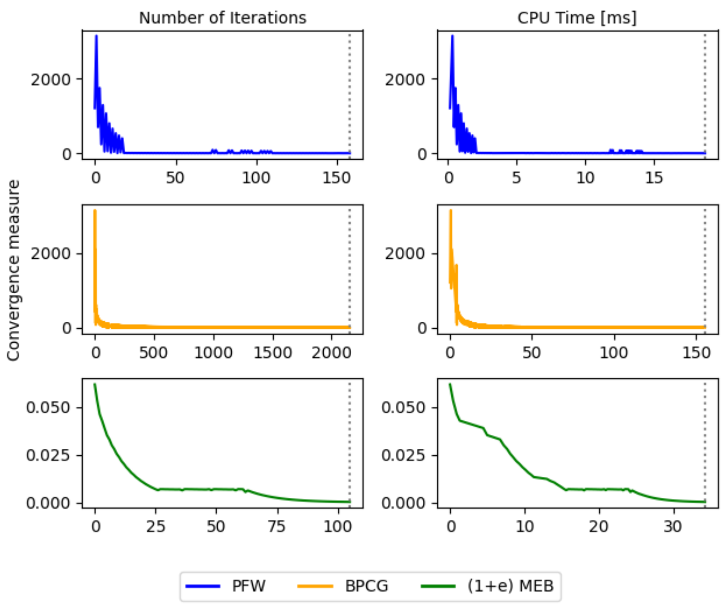

Figure 16: Convergence measure for Vertebral column dataset

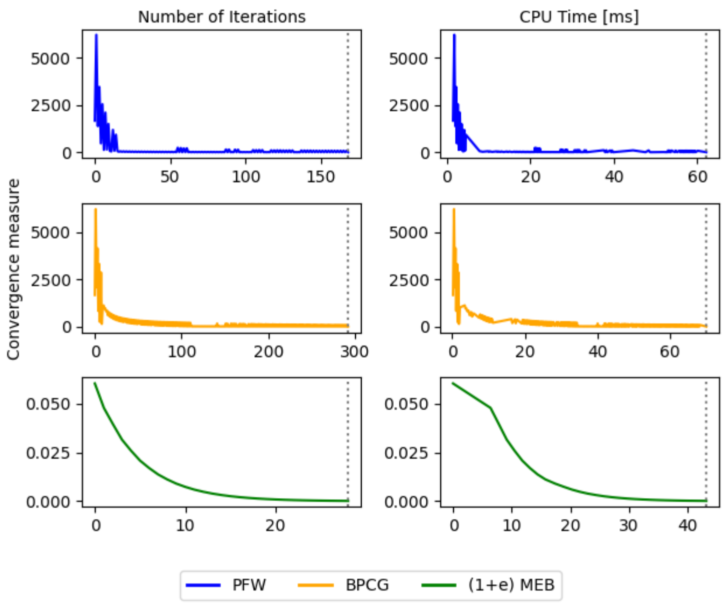

Figure 17: Convergence measure for Maternity risk dataset