Project 1 TMA4180

Student numbers: 767971, 767966, 767969

Wednesday 28th February, 2018

Question 1

For model 1, we solve our problem as the least squares problem by first defining residuals

$$r_i(A,c) := \begin{cases} \max\{(z_i-c)^T A(z_i-c)-1, 0\}, & \text{if } w_i > 0, \\ \max\{1-(z_i-c)^T A(z_i-c), 0\}, & \text{otherwise,} \end{cases} \tag{1}$$

and then we try to minimize the square of these residuals

$$\min_{x=(A,c)\in\mathbb{R}^{n(n+1)/2+n}} f(x) = \sum_{i=1}^{m} r_i(A,c)^2. \tag{2}$$

For model 2, we define the separating set differently, leading to a different definition of the residuals

$$\hat{r}_i(A,b) := \begin{cases} \max\{z_i^T A z_i + b^T z_i - 1, 0\}, & \text{if } w_i > 0, \\ \max\{1 - z_i^T A z_i - b^T z_i, 0\}, & \text{otherwise.} \end{cases} \tag{3}$$

Again we try to minimize the square of these residuals

$$\min_{x=(A,b)\in\mathbb{R}^{n(n+1)/2+n}} \hat{f}(x) = \sum_{i=1}^{m} \hat{r}_i(A,b)^2. \tag{4}$$

We now note that for both models, $r_i^2(x)$ can be written as

$$r_i^2(x) = [\max\{0, h(x)\}]^2, \tag{5}$$

where $h(x)$ is a smooth function. We then denote

$$g(\xi) = [\max\{0, \xi\}]^2, \tag{6}$$

and notice that f_i can be written as

$$f_i(x) = r_i^2(x) = g(h(x)), \tag{7}$$

and in this way, f is a sum of f_i.

We want to show that this function is continuously differentiable. The derivative of $f_i = g(h(x))$ will be

$$\frac{\partial f_i}{\partial x_i} = \frac{\partial g}{\partial \xi}(h(x))\frac{\partial h}{\partial x_i}(x). \tag{8}$$

The function $h(x)$ will in both models be a smooth function, so we only need to verify that $\frac{\partial g}{\partial \xi}(y)$ is a continuously differentiable function. Because g is the square of a max-function, it can be written on the following form

$$g(\xi) = \begin{cases} \xi^2, & \text{if } \xi > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

Now we have to verify that $g(\xi)$ is continuously differentiable. We do so by considering the point $\xi = 0$ and verify that the derivative from the left is equal that from the right. It is easily seen that from negative values of ξ, denote it 0^-, the derivative $g'(0^-) = 0$, becuase $g(\xi) = 0$ for $\xi < 0$. Now we compute the derivative from the positive direction.

$$g'(0^+) = \lim_{k \to 0^+} \frac{g(0+k) - g(0)}{k} = \lim_{k \to 0^+} \frac{k^2 - 0}{k} = 0 = g'(0^-)$$

Because the derivative is the same from both directions at $\xi = 0$, $g(\xi)$ is smooth elsewhere and $h(x)$ is a smooth function, $f_i = g(h(x))$ will be continuously differentiable. Finally, f is a function of f_i's and will therefore be continuously differentiable.

The function f will typically not be twice differentiable. This can be shown by computing the second derivative of $g(\xi)$ from both directions of zero. From the negative direction, $\xi < 0$, $g(\xi) = 0$ implies $g''(0^-) = 0$. However, from the positive direction $g(\xi) = \xi^2$ implies $g''(0^+) = 2$. These are not equal, and the function will therefore not typically be twice continuously differentiable.

Now we want to verify that the gradient of the function in equation (4) is Lipschitz continuous, i.e.

$$\|\nabla \hat{f}(y) - \nabla \hat{f}(x)\| \leq L\|y - x\|.$$

We use that $\hat{f}$ is a sum of f_i defined in equation (7). Each component of the gradient is defined in equation (8). Because $h(x)$ is a linear function of x, the term $\frac{\partial h}{\partial x_i}(x)$ will be constant, and we only need to prove Lipschitz continuity of the first term $\frac{\partial g}{\partial \xi}(\xi)$, i.e. there exists $x, y \in \mathbb{R}$ and $M \geq 0$ s.t.

$$\|\frac{\partial g}{\partial \xi}(y) - \frac{\partial g}{\partial \xi}(x)\| \leq M\|y - x\|.$$

We now have three different cases for y and x. First, if both $x, y \leq 0$, then the left hand side will equal to 0, and because $M \geq 0$ and $\|y - x\| \geq 0$ the inequality will be satisfied. Second case, $x \leq 0$ and $y \geq 0$. Now construct a new function

$$l(\xi) = \begin{cases} g'(\xi), & \text{if } \xi \geq 0, \\ -g'(|\xi|), & \text{if } \xi < 0. \end{cases}$$

Then $\frac{\partial g}{\partial \xi}(x) = 0 \geq l(x)$. Now, because $l(\xi)$ is linear, it is Lipschitz continuous. Also, because $\frac{\partial g}{\partial \xi}(y) = l(y) \geq 0$ and $l(x) \leq 0$, we get

$$\|\frac{\partial g}{\partial \xi}(y) - \frac{\partial g}{\partial \xi}(x)\| \leq \|l(y) - l(x)\| \leq M\|y - x\|.$$

Finally, when $x, y \geq 0$, the gradient of g is a linear function, which is Lipschitz continuous. We have shown all three cases, and therefore $\nabla f(x)$ will be Lipschitz continuous.

Question 2

The contours of $r_i(A, c)$ for Model 1 will be centered around c. Hence, if a point with positive weight is placed in the center of the ellipses, at c, there will be an infinite amount of globally optimal solutions. Any choice of a matrix A will minimize the problem since the residual will be zero. In the case of Model 2, the contours will be centered around the origin for positive weights. For instance, if there is only one positive point in the origin, there will be an infinite amount of choices for A and b that will yield an optimal solution to the problem.

The objective function is coercive if $f(\boldsymbol{x}) \to +\infty$ as $\|\boldsymbol{x}\| \to \infty$. For Model 1, consider the case of a positive point centered around c. As described, any choice of A would solve the problem globally since r_i would be zero. Letting the entries of A grow towards infinity, the objective function would still have a finite value, in this case zero. Hence the objection function is not coercive. In the case of Model 2, choose a positive point in the origin. The same reasoning applies, and any choice of A is valid. Letting the entries of A go towards infinity, the objective functions holds a finite value, and is therefore not coercive.

Question 3

We will now show that the objective function defined in equation (4) is convex.

Let $g : \mathbb{R}^l \to \mathbb{R}$ be a convex function and $h : \mathbb{R}^k \to \mathbb{R}^l$ be an affine function. That is, $h(x) = Ax + b$ for some $A \in \mathbb{R}^{l \times k}$ and $b \in \mathbb{R}^l$. Then we

denote the composition of g and h as $f(x) = g(h(x)) : \mathbb{R}^k \to \mathbb{R}$, and show that this composition is convex. Let $t \in [0, 1]$, and $x_1, x_2 \in \mathbb{R}^k$. Then,

$$\begin{aligned} f(tx_1 + (1-t)x_2) &= g(h(tx_1 + (1-t)x_2)) \\ &= g(A(tx_1 + (1-t)x_2) + b) \\ &= g(tAx_1 + (1-t)Ax_2 + tb + (1-t)b) \\ &= g(t(Ax_1 + b) + (1-t)(Ax_2 + b)) \\ &= g(th(x_1) + (1-t)h(x_2)) \\ &\leq tg(h(x_1)) + (1-t)g(h(x_2)) \quad \text{(because } g \text{ is convex)} \\ &= tf(x_1) + (1-t)f(x_2), \end{aligned}$$

so $f(x) = g(h(x))$ is convex when g is convex and h is affine.

To show that the objective function $\hat{f}(x) = \sum_i \hat{f}_i$, where $\hat{f}_i = g(h(x))$ is convex, we start by showing that the function g is convex.

Consider then the function on the form $g(\xi)$ as defined in equation (9). We have already proved that this function is continuously differentiable. Then the function is convex if and only if for every $x, y \in \mathbb{R}^n$ the inequality

$$g(y) \geq g(x) + \nabla g(x)^T (y - x), \tag{10}$$

is satisfied [1].

The gradient of g is

$$\nabla g(\xi) = \begin{cases} 2\xi, & \text{if } \xi > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

We have to consider four cases for (10). For $x, y \leq 0$ this inequality is trivially true since $g(y)$, $g(x)$ and $\nabla g(x)$ are zero in this domain. When $x \leq 0$ and $y \geq 0$ the equation becomes $y^2 \geq 0$, which is true. In the case of $y \leq 0$ and $x \geq 0$ we get

$$\begin{aligned} 0 &\geq x^2 + 2x(y - x) \\ 0 &\geq 2xy - x^2, \end{aligned}$$

which is true because y is zero or negative, and x is zero or positive, so $2xy$ will be zero or negative and $-x^2$ will be zero or negative.

Lastly, when $x, y \geq 0$ the equation becomes

$$\begin{aligned} y^2 &\geq x^2 + 2x(y - x) \\ (y - x)^2 &\geq 0, \end{aligned}$$

which is true. This means that the function g defined in equation (9) is convex.

We know that $h(x)$ is an affine function in model 2. In addition, g is convex. This means that the composition of these, $\hat{f}_i = g(h(x))$ will be convex by our earlier proof.

Finally, because the objective function $\hat{f}$ is a sum of convex functions $\hat{f}_i$, it will also be convex.

Question 4

We implement (1) and (3) as well as their derivatives (8) in Python. Comparing our gradient with a finite difference approximation, we get that error in the approximation decreases as $\epsilon \to 10^{-8}$ and increases somewhat after.

$$|\nabla f(x)^T \cdot p - \frac{f(x + \epsilon p) - f(x)}{\epsilon}| \geq 10^{-8}.$$

This is exactly what we'd expect considering that we are using double-floating point numbers, which are able to represent numbers up to around $1.1 \cdot 10^{16}$, and $10^{-16}/10^{-8} = 10^{-8}$.

Question 5

Implementation

To solve the unconstrained optimization problems (2) and (4) we implement steepest descent, BFGS and Fletcher-Reeves conjugate gradient search as described in [2, p. 21], [2, p. 140] and [2, p. 121], respectively. For steepest descent, we use backtracking linesearch as described in [2, p. 37]. For BFGS and Fletcher-Reeves we implement bisection linesearch as described in [2, p. 60] with the modification that in BFGS we do not require strong Wolfe. We implement BFGS with the modification that if the search direction becomes orthogonal to the gradient, we reset H to I. Fletcher-Reeves is implemented with the modification described by [2, eq. (5.48) p. 123]. We terminate when $\|\nabla f(x)\|_2 \leq 10^{-3}$ or if the number of iterations reaches 9999.

Figure explanation

In all following figures we run the algorithms on model 1 (left) and model 2 (right) for the given data points (z_k, w_k), marked by red ($w_k = -1$) and green ($w_k = 1$) dots. Above each figure is the number of iterations required to terminate (or 9999 if it did not) and the residual function value $f(x^*)$ for each of the algorithms. The ellipsoids for the algorithms are colored orange (steepest-descent), purple (BFGS) and yellow (Fletcher-Reeves). The algorithms will throughout the discussion be respectively referred to as SD, BFGS and FR.

Testing the implementation

We test our implementations in 2 dimensions by drawing $z_0, z_1 ... z_n$ from a uniform distribution on $\{z \in R^2 : x, y \in (-1.5, 1.5)\}$ with corresponding weights w with $P(w_i = 1) = P(w_i = -1) = 0.5$. We vary n throughout the discussion.

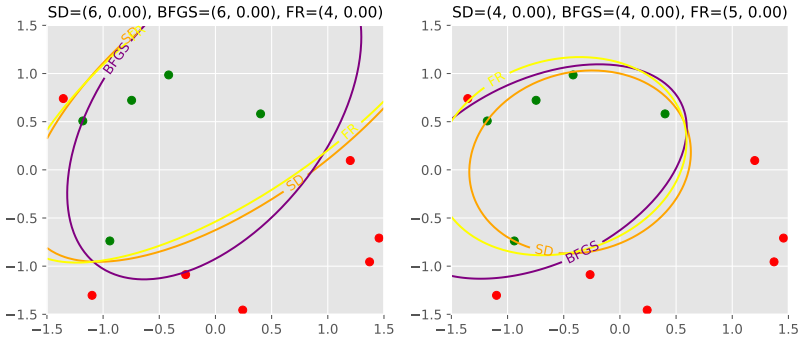

Figure 1

We observe from figure 1 that when all the positive points are grouped together in the middle, all the methods terminate in a small and equal amount of iterations for both models. When there exists some cone including $(0, 0)$ and the positive points, excluding the negative points, we can achieve the same residual function value with both models using a fraction of the computational work for model 2, see figure 2.

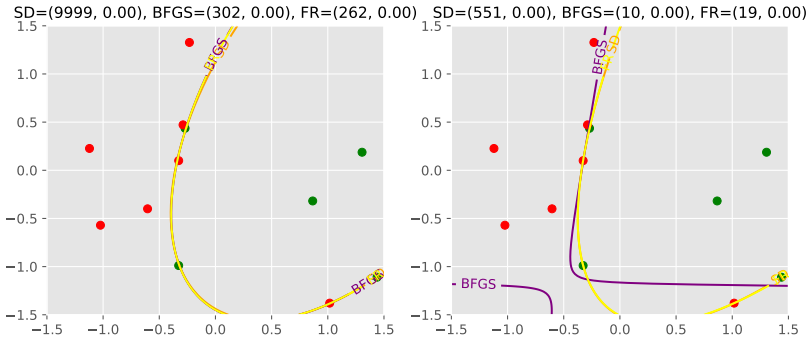

Figure 2

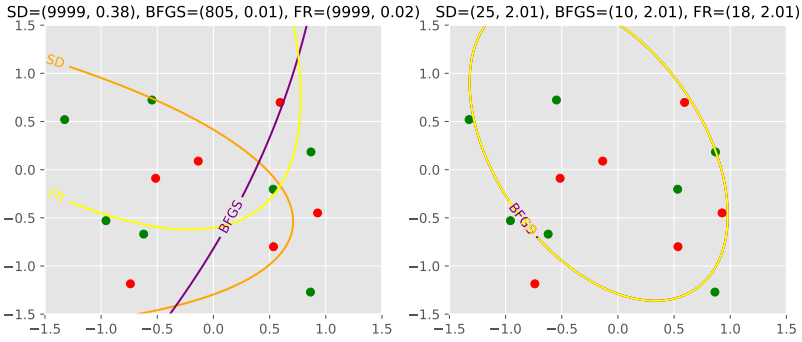

Figure 3

We observe that as the structure becomes increasingly complex both steepest descent and FR struggle with model 1. SD and FR terminate without reaching $\|\nabla f(x)\|_2 \leq 10^{-3}$, while BFGS requires 805 iterations. For model 2, the program terminates after a drastically lower amount of iterations, but the unyielding model results in a residual function value of $\hat{f}(x_2^*) = 2.01$, while model 1 attains $f(x_1^*) = 0.01$. The indices indicate that the local minimum of model 1 is not the same as for model 2. In general, if the positive points are not in some way centered around $(0, 0)$, model 2 will be bounded below, which we here see evidence of.

Choosing our points uniformly, we observe that as a general rule SD requires more iterations than FR which requires more than BFGS. For model 1, FR and SD exhibits a tendency to get stuck in saddle points, while BFGS does not suffer from this.

Tests

We will now consider our models' ability to find the structure of a "hidden" set; Given a sequence of samples (z_k, w_k) drawn from a set S, how well can we fit our models $f_{A,c}(x)$ and $\hat{f}_{A,b}(x)$ to that set, and at what speed can our methods (SD, BFGS and FR) do this.

Modelling ellipsoidal sets

We first consider sets that are constructed in the form (A, c) / (A, b), i.e. given enough data points, our models should be able to fully fit the underlying set. We choose $x = (1, 0, 1, 0.5, 0.5)^T$,

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad c = b = (0.5, 0.5)^T$$

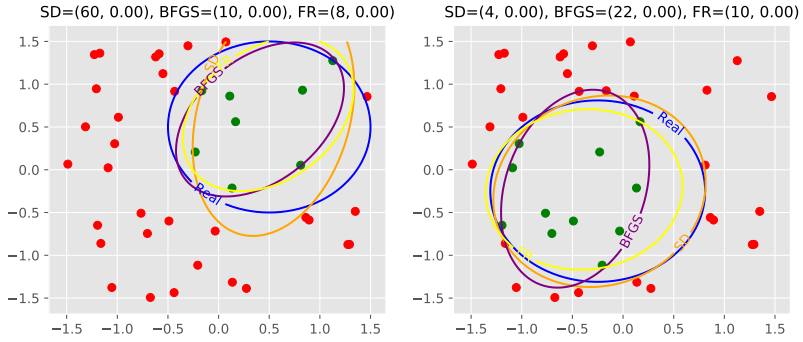

Figure 4

We now choose $x = (1, 0, 10^2, 0, 0)^T$,

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 10^2 \end{pmatrix}, \quad c = b = (0, 0)^T$$

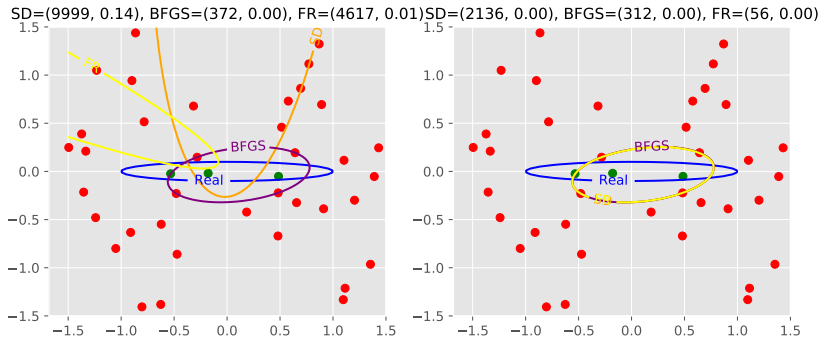

Figure 5

We see that our algorithms, given that there exists several local minima, have a tendency to choose $f(x^*)$ with the smallest $\|x\|$. We can restrict this somewhat by increasing our amount of data-points, see figure 6, but we still observe that SD and BFGS struggle in solving for a very oblong ellipsoid. Notice that in this case FR achieves a sufficiently minimal solution in surprisingly few iterations. This indicates that in the case that our problems (1) and (3) have a somewhat smooth structure (as is the case when we increase our amount of data points), FR gives good results.

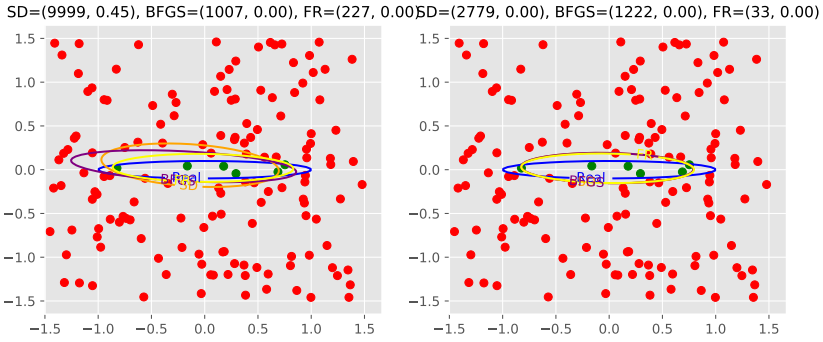

Figure 6

Modelling a square set

We then consider the square set, which is impossible to model perfectly in the form (A, c) / (A, b).

$$\begin{aligned} w_i &= 1, \quad \|z\|_\infty \leq 1 \\ w_i &= -1, \quad \text{otherwise} \end{aligned}$$

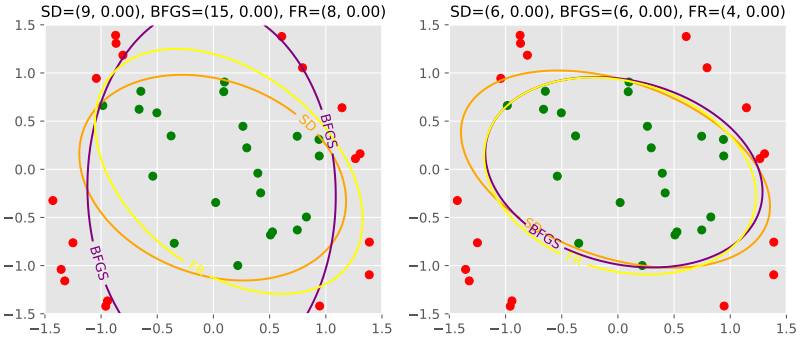

Figure 7

In figure 7, we see our models provide good approximations of the square set, and the search methods converge to the optimal model for the given data points in a small amount of iterations.

Modelling a square set with some misclassification

Building on the previous example, we can introduce some error of misclassifying the weights w_i. We set this to 0.10. This gives the following result.

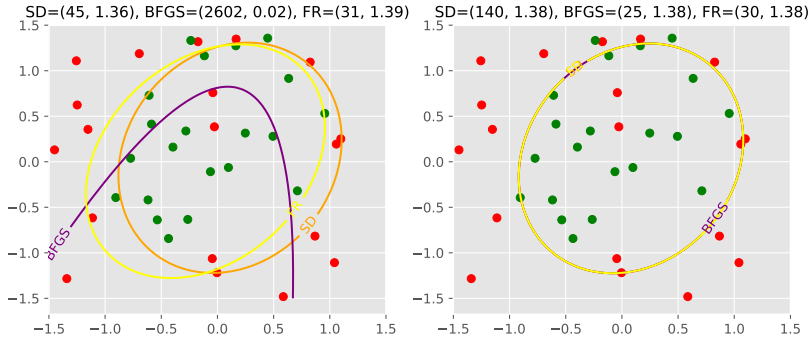

Figure 8

By figure 8, we see that the flexibility in model 1 lets BFGS 'overfit' our data and requires 2602 iterations to do this. This tendency disappears as the amount of data points becomes large, see figure 9.

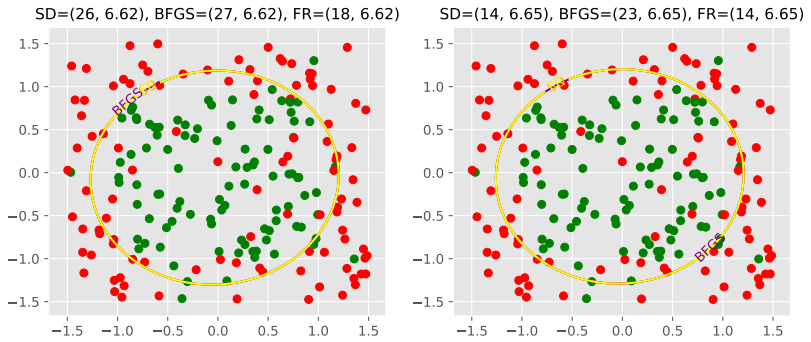

Figure 9

Extensions to 3D

We can extend the discussion to 3 dimensions. The algorithms are of course general in N dimensions, and ellipsoids in 3D correspond to $x = (a_1 \dots a_6\, c_1 \dots c_3)^T \in \mathbb{R}^9$. Again we choose (z_k, w_k) at random and let the different methods find the optimal solutions to model 1.

Model 1

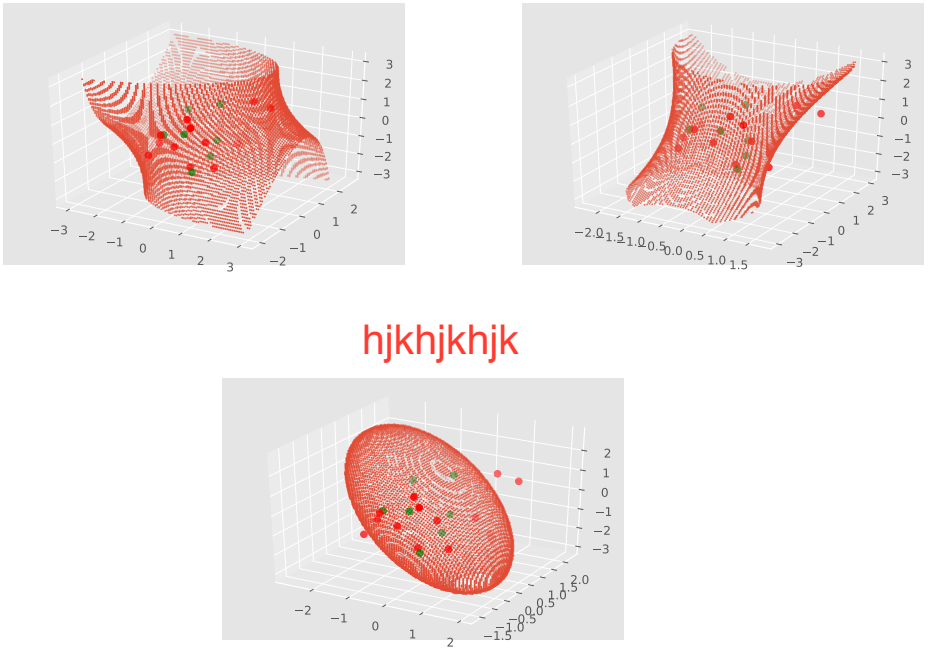

Figure 10: Left to right: SD=(305, $3.30 \cdot 10^{-7}$), BFGS=(17, 0.0), FR=(20, 0.0)

Model 2

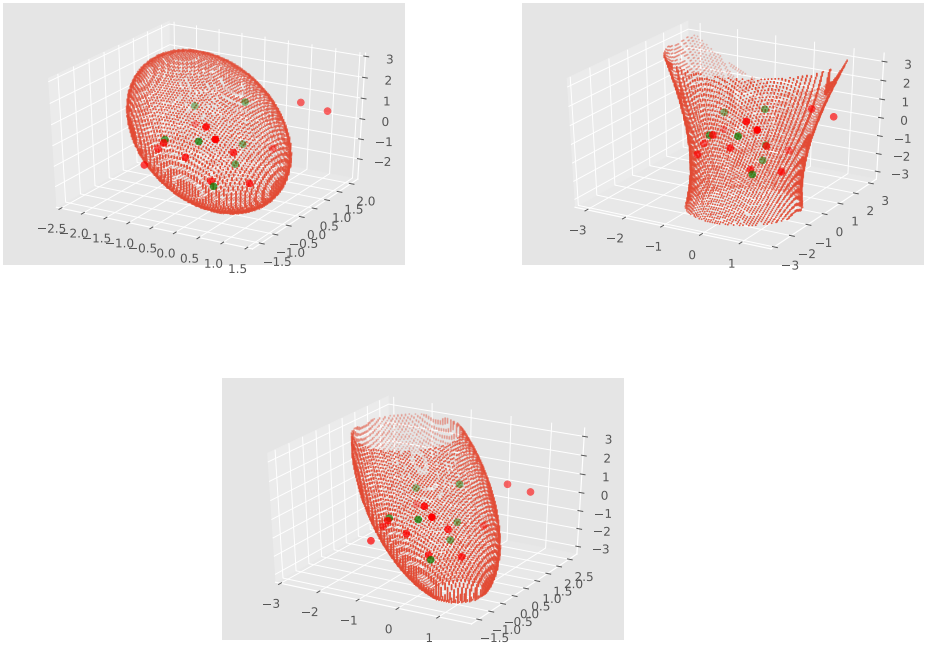

Figure 11: Left to right: SD=(288, $1.03 \cdot 10^{-6}$), BFGS=(6, 0.0), FR=(8, 0.0)

As plotting 3D contours of 4D functions is at a really early stage in matplotlib, we restrict this section to the empirical evidence that the methods converge in a 3-dimensional setting, see figure 10 and 11, and assume that any instability experienced in 2D will also be experienced in 3D.

References

[1] Grasmair, Markus. *Basic Properties of Convex Functions*

[2] Jorge Nocedahl, Stephen J. Wright. 2nd edition, 2000. Springer. *Numerical Optimization*