Convex Optimization - The Diet Problem

Francesco Bardi, Maurice Amendt, Luca Zampieri
Convex Optimization and Applications, EPFL, Switzerland

***Abstract*—The goal of this paper is to demonstrate our comprehension of a specific topic seen during the class of Convex Optimization.**
We firstly present a multi-criterion version of the diet problem that could be applied in the context of restoration to innovate current ideas. Secondly we solve it for a set of parameters. And finally we make a study of convergence for the Interior Point Method, and compare several methods for solving linear problems. We then show our implementation of a particular Interior Point Method algorithm. Finally we solve the problem for a set of parameters interpreting the obtained results with a focus on sensitivity analysis.

I. Introduction

Convex Optimization is a branch of optimization in which a convex function is optimized over a convex domain. Since the 1990's it has gained in popularity due to the discovery of efficient algorithms to solve such problems, making the main distinction between convex vs non-convex optimization instead of the usual linear vs non-linear. The main characteristic of convex problems is that local minima turn out to be global ones.
The problem we are going to analyze in the present work falls under a special category of convex optimization, named Linear Programming. As already indicated by its name, the objective function, as well as the constraints, are linear for that class of problems. In all generality, such a problem can be stated as:

$$\begin{aligned} &\text{min: } c^\top x \\ &\text{s.t.: } Ax \leq b \end{aligned}$$

The vector c is called the objective function, A is known as the constraint matrix. Although the formulation of such problems is rather simple, astonishingly many practical issues can be stated as an LP, as for instance network flow or transportation problems. As of today, LP's can be solved very efficiently and many software tools are available, solving them using various methods.
In the present report, the problem of interest is known as the diet problem. The diet problem is a famous LP problem due to its historical significance, instructional value, and ease of understanding. The problem has first been tackled by George Stigler in the early 40's. George Stigler was interested in how much an average man had to eat to satisfy his daily nutritional needs, whilst keeping the cost of the food as low as possible. Due to his work, the diet problem is likewise known as the Stigler diet.

II. The Diet problem

The original diet problem is the following:
$Minimize$: the cost of food eaten during one day
$Subject\ to$: the requirements that the diet satisfy a person's nutritional requirements and that not too much of any one food be eaten.

To formulate the problem mathematically we introduce the vector x representing the amount of food of each ingredient and a matrix A, that contains the nutritional values of each ingredient. We then desire to constrain the quantity of each nutriment for the diet setting a vector of lower bound for each nutriment b_1 and similarly an upper bound b_2. We can write the problem as follows:

$$\begin{aligned} &\text{min: } c^\top x \\ &\text{s.t.: } b_1 \leq Ax \leq b_2 \\ &\qquad x_i \geq 0 \qquad \forall i \end{aligned}$$

where we notice that both the objective and the constraints are linear thus the diet problem is indeed an LP problem.

III. Revisited Diet Problem

In order to make the problem more interesting, we consider the following modification.

A. Context

Suppose we want to launch a new concept: a restaurant that aims to give the optimal user experience for a minimal cost. In our little restaurant we offer several meals for which we have a set of nutriments, and cost for 100g of each meal. In an app, we then invite the client to enter his preferences for a set of food and thanks to a recommendation system we predict the likeliness that the client has for each meal (one can imagine it as the Netflix recommendation system). The client has his own account, and the more he comes the more accurate the predictions will be. The menu then changes every day to create diversity, while still suggesting something tasteful to the client.
Supposing that we already have the recommendation system (which is beyond the scope of this course, but which we would know how to implement from other courses), the problem can be solved reformulating the diet problem and adding the "taste" variable. We end up with a multi-criterion objective optimization problem: maximize the taste while minimizing the cost, subject to the set of constraint such as

the minimum and maximum amount of food we want to eat, maximum fat, proteins and carbohydrates.

B. Mathematical formulation

Introducing the trade-off parameter $\gamma \geq 0$ and calling the number of dishes n, we can write the problem as follows:

$$\begin{aligned} \text{min: } & -t^\top x + \gamma c^\top x \\ \text{s.t.: } & b_1 \leq Ax \leq b_2 \\ & food_{min} \leq \sum_{i=1}^{n} x_i \leq food_{max} \\ & x_i \geq 0 \ \forall i = 1:n \end{aligned}$$

Where t is the taste vector, c the cost vector, x_i the amount of each food, A the matrix of food characteristics (includes nutriments, costs and tastes), b_1 and b_2 the lower and upper bound vectors for the constrains, and $food_{min}$ and $food_{max}$ the minimum and maximum amount of food.
We solve for a variety of different $\gamma > 0$ and will end up with the set of pareto-optimal values. The trade-off parameter γ represents how much importance is given to the taste w.r.t. the cost. The user can slide through the pareto-optimal values and chose the trade-off that suits its desires.

Note that the previous problem can be reformulated as follows:

$$\text{min: } \widetilde{c}^\top(\gamma)x \tag{1}$$
$$\text{s.t.: } \widetilde{A}x - \widetilde{b} \leq 0 \tag{2}$$

Where:

$$\begin{aligned} \widetilde{c}(\gamma) =& -t^\top + \gamma c^\top \\ \widetilde{A} =& [A, -A, \mathbf{1}, -\mathbf{1}, -I_{nxn}]^\top \\ \widetilde{b} =& [b_2, -b_1, food_{max}, -food_{min}, \mathbf{0}_n]^\top \end{aligned}$$

Calling K the number of rows in matrix A and n the number of dishes, $\widetilde{A}$ will have $m = (2K + 2 + n)$ rows.
Which is the standard formulation for a linear problem with inequality constraints, with the exception of the γ-dependence of $\widetilde{c}$.
Note that all optimal points found by scalarisation are pareto-optima, but the converse is not always true. In our case, since the problem is convex we should be able to find all pareto-optimal point by the use of scalarisation.

IV. Dual problem

To each problem p we can associate a dual problem d. The optimal value d^* gives a lower bound on the optimal value of our original problem $p^* = p(x^*)$ which is called **Weak Duality**. Moreover, the objective being linear (thus convex), and all the constraints being linear as well, if x^* is feasible the **Slater's Conditions** ensures **strong duality**: $p^* = d^*$

To get the dual, we start by writing down the Lagrangian of problem (1) and, dropping the tilde:

$$\mathcal{L}(x, \lambda) = -t^\top x + \gamma c^\top x + \sum_{i=1}^{n}\sum_{j=1}^{m} \lambda_i (a_{ij} x_j - b_j) \tag{3}$$

with a_{ij}, b_j entries of the respective matrix A and vector b; n the number of different dishes; m the total number of constraints. To then get $g(\lambda) = inf_{x \in \mathcal{D}}(\mathcal{L}(x, \lambda))$:

$$g(\lambda) = \begin{cases} -b^\top \lambda, & \text{if } (-t + \gamma c + A^\top \lambda) \\ -\infty, & \text{otherwise} \end{cases} \tag{4}$$

and our dual problem is:

$$\begin{aligned} &\text{Maximize } g(\lambda) \\ &\text{s.t.: } (-t + \gamma c + A^\top \lambda) = 0 \\ &\quad \lambda_i \geq 0 \quad \forall\ i = 1:n \end{aligned}$$

V. The barrier method

Interior point methods have revolutionized the way to solve convex optimization problems. We show here how to solve problem 1 using the barrier method which is a particular interior-point algorithm. The optimal solution is obtained applying Newton method to a sequence of problems in which the inequality constraints are made implicit in the objective function through an approximated indicator function. In our case we used the logarithmic barrier function so that the approximated problems $\mathcal{P}(t)$ are:

$$minimize \qquad t f_0(x) + \phi(x)$$

with:

$$\begin{aligned} f_0(x) &= \tilde{c}^T x \\ \phi(x) &= -\sum_{i=1}^{i=m} log(-f_i(x)) = -\sum_{i=1}^{i=m} log(\tilde{b}_i - \tilde{a}_i^T x) \end{aligned}$$

and where $\tilde{a}_i^T$ are the rows of $\tilde{A}$. The new set of problems $\mathcal{P}(t)$ are now unconstrained optimization problems which can be solved using the Newton method (if $\tilde{A}$ has rank n). For each problem the optimal solution $x^*(t)$ must satisfy:

$$\begin{aligned} &\nabla f_0(x^*(t)) + \nabla \phi(x^*(t)) = \\ &\nabla f_0(x^*(t)) - \sum_{i=1}^{i=m} \frac{1}{f_i(x^*(t))} \nabla f_i(x^*(t)) = 0 \end{aligned}$$

which is equivalent to say that $x^*(t)$ minimizes the Lagrangian (3) with

$$\lambda_i^* = -\frac{1}{f_i(x^*(t))}$$

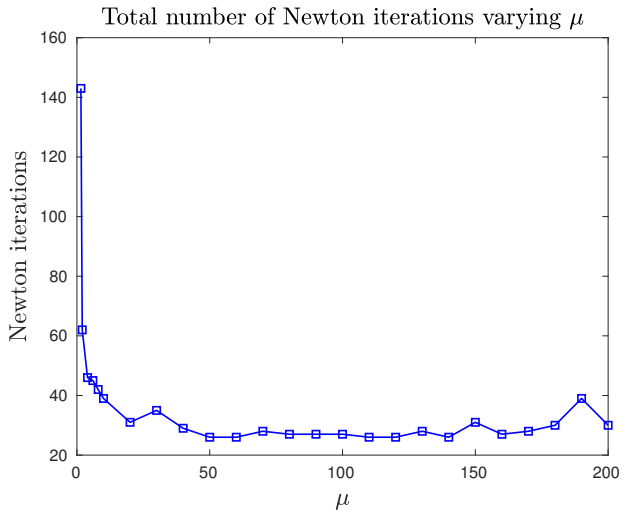

Fig. 1. Cumulative number of Newton iterations for different values of μ

Therefore the following holds for the dual function:

$$\begin{aligned} g(\lambda^*) &= f_0(x^*(t)) + \sum_{i=1}^{i=m} \lambda_i^* f_i(x^*(t)) \\ &= f_0(x^*(t)) - m/t \\ &\leq p^* \end{aligned}$$

showing that $x^*(t)$ converges to the optimal value p^* as $t \to \infty$.

VI. MATLAB IMPEMENTATION

We have implemented the barrier method in MATLAB. Starting from an initial feasible solution x_0 and $t = 1$ we solve $\mathcal{P}(t)$ using the Newton method with backtracking line search. The obtained optimal point is used as the initial value for a new problem $\mathcal{P}(\mu t)$ where $\mu > 1$. The procedure is iterated until a desired tolerance ϵ on the duality gap is reached. The pseudo code is given in 1.

Data: $x_0 \in \mathcal{D}(f)$, ϵ
Result: optimal value x^*
while $m/t > \epsilon$ **do**
  solve $\mathcal{P}(t)$ using the Newton method with initial value x_0;
  update $x_0 := x^*(t)$;
  increase $t := \mu t$;
end

Algorithm 1: Barrier method algorithm

In order to verify our implementation we solved the same LP which is solved in Boyd textbook [1]. Figure 1 shows the cumulative number of Newton iterations for different values of μ and shows that the performances of our algorithm totally agree with Boyd's implementation.

We then proceeded to solve problem 1 for different values of γ both with the implemented algorithm and with python. In the following we report only the results obtained with python but both implementations are available in the GitHub repository [5].

VII. AN APPLICATION

A. Statement

To illustrate what has been said above, lets consider the following dishes with associated nutritional values [4]:

Food	Cost	Calories	fat	carbohydrates	proteins	taste
Fried Fish	5.5	199	11.37	6.71	16.72	4.9
meatballs	4.5	191	13	5.2	13	3.9
octopus salad	5	128	5.5	5.5	14	4.4
Meat Ravioli	3.5	191	7.43	17.36	12.58	4
Tomato Pasta	3.4	116	1.88	21.04	4.14	4.4

Now suppose we absolutely want to pay less than 14$. Moreover our profile suggests less than 450 calories, less than 30g of fat, less than 50g of carbohydrates, less than 60g of proteins and more than 30g of proteins. Finally we want to eat at least 80g but less than 300g of food.

We thus solve problem (1) with:

$$\begin{aligned} \widetilde{c}(\gamma) =& -t^\top + \gamma c^\top \\ \widetilde{A} =& [A, -A, \mathbf{1}, -\mathbf{1}, -I_{nxn}]^\top \\ \widetilde{b} =& [b_2, -b_1, food_{max}, -food_{min}, \mathbf{0}_n]^\top \end{aligned}$$

where:

$$t^\top = \begin{bmatrix} 4.9 & 3.9 & 4.4 & 4 & 4.4 \end{bmatrix}$$

$$c^\top = \begin{bmatrix} 5.5 & 4.5 & 5 & 3.5 & 3.4 \end{bmatrix}$$

$$A = \begin{bmatrix} c_1 & 199 & 11.37 & 6.71 & 16.72 \\ c_2 & 191 & 13 & 5.2 & 13 \\ c_3 & 128 & 5.5 & 5.5 & 14 \\ c_4 & 191 & 7.43 & 17.36 & 12.58 \\ c_5 & 116 & 1.88 & 21.04 & 4.14 \end{bmatrix}$$

$$b_1^\top = \begin{bmatrix} 14 & 450 & 30 & 50 & 60 \end{bmatrix}$$

$$b_2^\top = \begin{bmatrix} 0 & 0 & 0 & 0 & 30 \end{bmatrix}$$

and $food_{min} = 0.8$, $food_{max} = 3$

B. Results

Solving (1) with python[5] using cvxopt[6] we get the trade-off illustrated in figure 2.

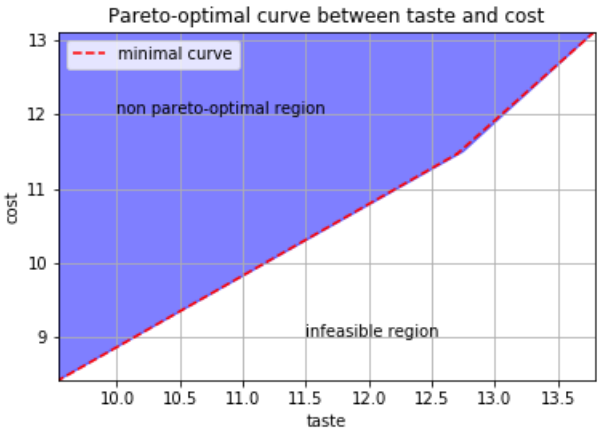

Fig. 2. Pareto optimal solutions for our set of food

We observe that the solution is at the border of the domain, as it should be in linear programming. Moreover, as expected from dual discussion above, the primal optimal objective and the dual optimal objective are the same (i.e the dual gap is 0): we have strong duality!

The pareto-optimal solution results in the combination of dishes seen in figure 3. The interesting changes of distributions being in the range $\gamma \in [0.6, 1.2]$.

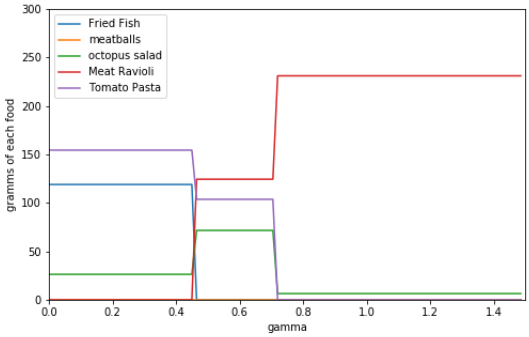

Fig. 3. Pareto optimal distribution of dishes

We recall that small values of γ prioritize the maximization of the taste while a big γ prioritizes a reduction of the cost. As we see, for small γ we can afford some fried fish, the most expensive meal, whereas for larger gamma we switch to meat ravioli that are the cheapest after the pasta. Pasta is not chosen to minimize the cost probably because it does not fulfill the constraint of the minimum amount of proteins, but lets dig into the sensitivity analysis to draw more conclusions.

C. Sensitivity analysis

Reminder: if we perturb the original problem by a **small** perturbation u i.e our constraints are now written as $\widetilde{A}x - \widetilde{b} \leq \mathbf{u}$. Since strong duality holds for the unperturbed problem we can write: $p^*(u) \geq p^*(0) - u^\top \lambda^*$. Thus for large λ_i there will be a significant increase on $p*$ if we tighten constraint u_i but almost none if λ_i small.

We now look at the dual variables to proceed to the sensitivity analysis. We have 17 constraints on the primal problem, thus 17 dual variables. Of these, only five have significant values and they are represented as a function of γ in figure 4.

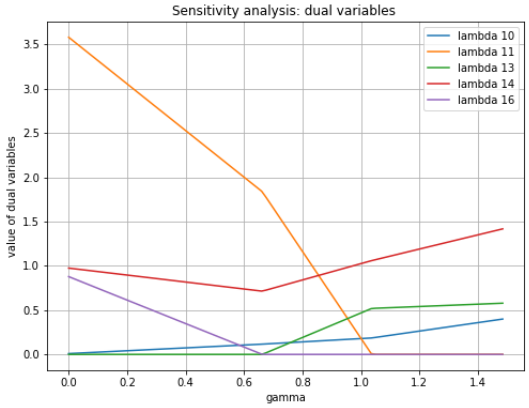

Fig. 4. The five largest dual variables as a function of the trade-off parameter γ. λ_{10} is the constraint on the minimum amount of proteins (30g), λ_{11} is the constraint on the maximum amount of food (300g), and $(\lambda_{13}, \lambda_{14}, \lambda_{15})$ are the constraint imposing a non-negative amount for each dish.

We infer from the figure that the objective function is very sensible to λ_{11} for small γ i.e. that if we decrease the total amount of food we can eat we will increase the objective value. This makes perfect sense since our measure of taste is, in our model, multiplied by the quantity of food eaten, and increasing the objective would mean decrease the overall taste for small gamma. We also see that the bigger the γ (i.e. cost is getting more important than taste) the more sensible the problem becomes to the constraint on the total amount of proteins (λ_{10}). This is explained by the fact that tomato pasta is the cheapest but contains a very small amount of proteins. This forces us to prefer meat ravioli (c.f. fig 3) even though they are slightly more expensive.
Finally, $\lambda_{13}, \lambda_{14}, \lambda_{15}$ don't give us much information because we will not be able to have negative amount of food, and small variation of the other constraints should not influence our solution since their dual variable is close to 0.

VIII. Improvements and Conclusion

Due to its early statement in the 1940's, the diet problem has often served as a first introductory example of LP. However, due to its simplistic approach, a straightforward and basic formulation of the problem is barely ever practical, as the result can often be very monotonous and thus not very palatable as a human diet.

Consequently, a more general formulation of the famous diet problem was considered in the present work. In fact, human satisfaction has been taken into account, by also considering the taste of different dishes, resulting in a multi-criteria optimization problem. The Pareto-optimal points for a given instance were computed by solving the linear programme, obtained by introducing a trade-off parameter between the taste and the cost of the dishes. A sensitivity analysis was performed via duality.

The generalization of the diet problem as a multi-criteria optimization problem, brings some interesting flexibility. While, for instance, young people could be more interested in diets prioritizing the cost, more well-off people could opt for more tasteful dishes at the expense of cost.

Another interesting generalization is when the entries of the constraint matrix A are not deterministically known. This situation corresponds to slight uncertainties or variations in the nutritional values of the ingredients, due to, for instance, imperfect production or seasonal changes. More precisely, if the nutritional values are supposed to lie in some ellipsoid, the problem can be formulated as a second-order cone programme (SOCP) an solved efficiently by appropriate tools.

REFERENCES

[1] Stephen Boyd, Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press. 2004

[2] The Diet Problem, Author(s): George B. Dantzig

[3] The Diet Problem: A WWW-based Interactive Case Study in Linear Programming (http://www.mcs.anl.gov/home/otc/Guide/CaseStudies/diet/) authors: Joseph Czyzyk Timothy J. Wisniewski

[4] Links to the nutritional value of our five dishes:

- https://www.fatsecret.it/calorie-nutrizione/generico/pesce-fritto-in-pastella
- https://www.fatsecret.it/calorie-nutrizione/ikea-food/polpette-di-pollo/100g
- https://www.fatsecret.it/calorie-nutrizione/generico/ravioli-ripieni-alla-carne
- https://www.fatsecret.it/calorie-nutrizione/generico/pasta-al-sugo
- https://www.fatsecret.it/calorie-nutrizione/fiorital/insalata-di-polpo/100g

[5] https://github.com/LucaZampieri/Convex_opti2018/blob/master/Convex%20project.ipynb

[6] http://cvxopt.org/