Interior Point Method for Linear Programs
CS544-HW3

Ziyang Xie, Yufeng Liu, Jiangwei Yu

1 Introduction

This study shows the performance of two kinds of interior point solvers for linear programs. Specifically, we explored and compared the central path method and primal-dual methods on linear problems in standard form. The problem scale ranging from 30 variables and 10 constraints to 200 variables and 100 constraints. Our Code is available here: `https://github.com/ZiYang-xie/CS544-24Spring/tree/main/mp3`

2 Problem Formulation

In this project, we consider optimizing the linear program of the standard form:

$$\min\ \mathbf{c}^\top \mathbf{x} \quad s.t.\ \mathbf{A}\mathbf{x} - \mathbf{b} = \mathbf{0},\ \mathbf{x} \geq \mathbf{0}$$

where $\mathbf{c}, \mathbf{x} \in \mathbb{R}^n,\ \mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{b} \in \mathbb{R}^m$, n is the variable number and m is the constrain number. In our case, we set $\mathbf{c}$ to only contain random positive values. This method ensures that the lowest possible value for our problem is zero, which makes the problem workable and prevents the lowest bound from being negative infinity. $\mathbf{A}$ and $\mathbf{b}$ are random generated.

3 Method

3.1 Central Path Method

The central path method is used in optimization problems where the goal is to find the maximum or minimum value of a function subject to certain constraints. These problems can be linear or nonlinear, but the method is particularly significant in linear programming. The central path is a trajectory that connects feasible solutions from an interior point method to the optimal solution, while strictly maintaining inferiority (i.e. $x \geq 0$). The central path is defined for $t > 0$ and approaches the boundary of the feasible region as t approaches infinity.

To follow the central path, a barrier term is added to the original objective function, creating a modified objective function that includes both the original linear objective and a barrier function to prevent the solution from reaching the boundary of the feasible region:

$$\Phi(x; t) = -\sum_i \log(-g_i(x)), \quad s.t.\ g_i(x) \leq 0$$

In our case, $\Phi(x; t) = -\sum_i \log(x)$, which indicates when $x \to 0^+$ then $\phi(x; t) \to \inf$ to prevent the solution go out of the feasible region. To satisfy Karush–Kuhn–Tucker(KKT [2]) complementary slackness, given $\lambda_i g_i = -\frac{1}{t}$, we need to gradually increase t (e.g. $t_{i+1} = \rho t_i,\ \rho > 1$) during optimization.

With barrier function, our problem can be transformed into the following form with linear constraints:

$$\min\ tf(x) + \Phi(x), \quad s.t\ Ax = b$$

where $f(x) = c^T x$, then we choose to use the augmented lagrangian method (ALM [1]) to further solve the optimization problem with equal constraints.

3.2 Full-Primal Dual Method

The primal-dual method starts with feasible solutions for both the primal and the dual problems. The form of the dual problem can be expressed as:

$$\max\ b^T y \quad s.t.\ A^T y + s = c,\ s \geq 0$$

where $y \in \mathbb{R}^m$ and $s \in \mathbb{R}^n$. It then iteratively updates these solutions based on the gradient information from the objective functions and the constraints of both problems. The updates are designed to improve the objective values of both the primal and the dual problems simultaneously. KKT conditions as follows:

$$A^T y + s = c, \tag{1}$$
$$Ax = b, \tag{2}$$
$$x_i s_i = 0,\ i = 0, 1, 2, \ldots n, \tag{3}$$
$$(x, s) \geq 0. \tag{4}$$

In the Full Primal-Dual method, instead of using predefined t to measure complementary condition, we introduce duality measurement μ as a convergence criterion:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i s_i = \frac{x^T s}{n}$$

The duality gap directly measures the difference between the primal and dual objective values, providing a clear and direct indicator of how close the current solutions are to optimality. A smaller duality gap signifies that the primal and dual solutions are converging towards the optimal solution, which directly correlates with the primary goal of optimization.

We can then compute the Jacobian matrix of the system and apply Newton method to solve the problem. The system of equations for the Newton step in the Full-Primal-Dual method is represented as follows:

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{bmatrix} = \begin{bmatrix} -r_c \\ -r_b \\ -XSe + \sigma \mu e \end{bmatrix}$$

In this representation: Δx, $\Delta \lambda$, and Δs are the search directions for the primal variables, dual variables, and slack variables, respectively. r_c, r_b, and $XS - e$ are the residuals of the primal, dual, and complementarity conditions, respectively. σ is a parameter controlling the centering and convergence of the algorithm.

By solving this system, the Full-Primal-Dual method iteratively updates the primal and dual variables, along with the slack variables, to navigate the path of steepest descent towards the point of optimality, efficiently reducing the duality gap and enhancing the overall convergence speed of the optimization process.

4 Questions

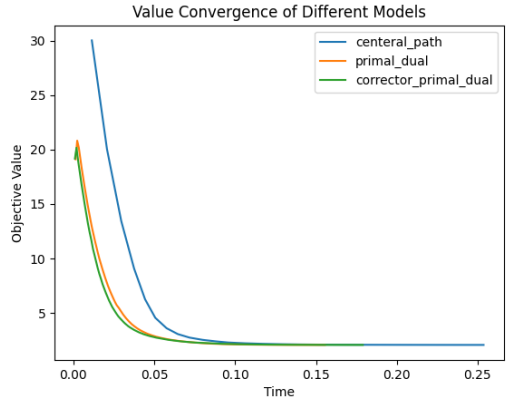
4.1 How bad is it to simply follow the central path compared to full primal-dual method?

We compared the central path method against the primal-dual method with the corrector step. The central path is implemented using Augmented Lagrangian Method [1] with L-BFGS [3]. Our findings are as follows:

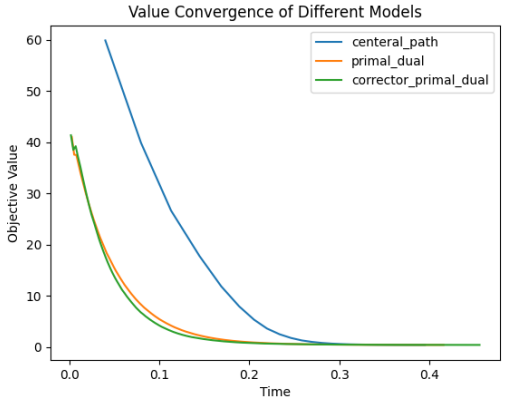
1. The central path relies heavily on the selection of hyper-parameters: update rate of t, the μ value(μ is the coefficient of the penalty term in ALM), and the numerical optimization method, while the primal-dual method is more robust and stable. We reasoned that since the central path method involves linear terms, quadratic terms, and logarithmic terms, the weights on those terms can heavily influence the performance. For example, a small μ can make the model ignore the linear constraints. Gradient descent suffers from exploding gradient when the barrier function is too close to the boundary. The primal-dual method on the other hand is robust in all kinds of inputs.

2. The primal-dual method is more efficient than the central path in small-scale problems. This is because the Central Path method leverages a bi-level optimization, every time it updates t, it will solve an ALM optimization which costs a lot of time. In contrast, Primal-Dual methods directly optimize all variables together which will be more efficient. When the complexity of problems scales up, it will pose difficulty to Primal-Dual methods and bi-level optimization will perform better. We will explain this more detail in 4.4

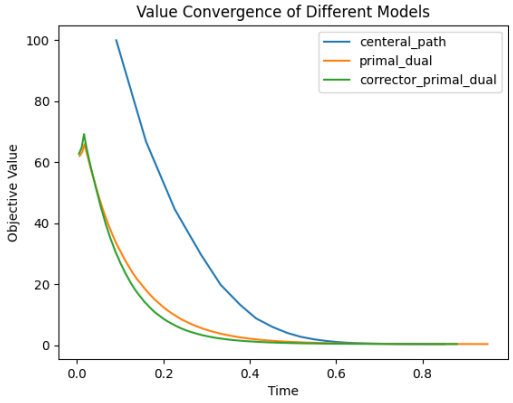
We conduct concrete experiments to support our conclusions in the following. Firstly, we compare the performance of these three methods in problems of different scales as shown in Figure 1. We conduct tests across four distinct scenarios, each with an increasing number of variables: 30, 60, and 100, respectively. Furthermore, we extend our examination by incrementally increasing the number of linear equation constraints from 10 to 30, thereby broadening the scope of our assessment to include more complex problem configurations. The findings reveal that, across all scenarios, the central path method demonstrates superior convergence speed compared to both the primal-dual method and its variant incorporating a corrector. Notably, the performance gap between the primal-dual and its corrector-enhanced counterpart is minimal, indicating their similar efficacy in handling the specified problem sets.



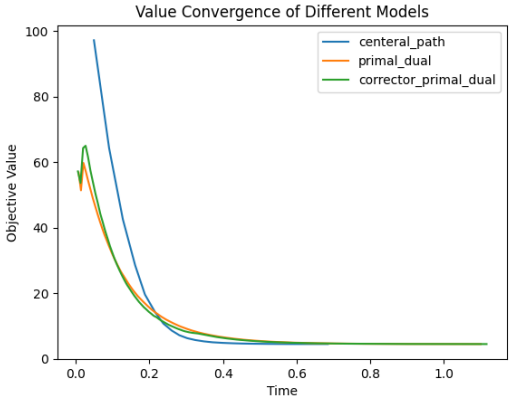
(a) num_var=30, num_eq=10



(b) num_var=60, num_eq=10



(c) num_var=100, num_eq=10

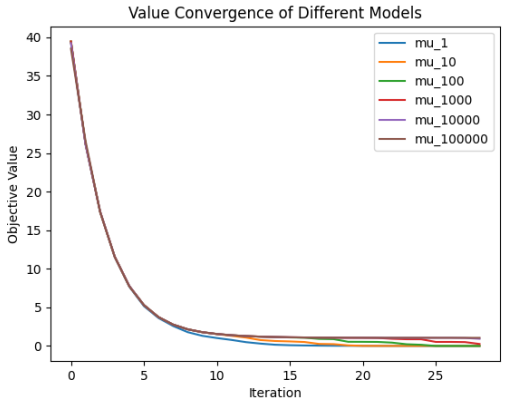


(d) num_var=100, num_eq=20

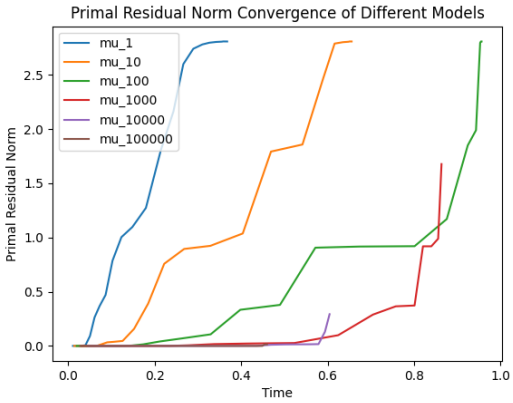
Figure 1: Comparision between Central Path, Primal-Dual, and Primal-Dual with corrector on problems at different scales

To show the central path method's sensitivity to the selection of hyper-parameters, we compare the performance with μ value. Figure 2(a) shows the optimization curve with different initial μ, and Figure 2(b) shows how far the solution deviates from the constraints, which is $Ax - b$ (The better the solution satisfies the constraints, the closer the value of Ax-b should be to 0). It is obvious that the solution is very sensitive to the selection of μ: if μ is too small, the optimizer may leave out the linear constraints and generate an infeasible

solution; if μ is too large, the optimizer may too focus on the constraints and thus get a worse solution. A general principle to set the initial value of μ is to set it comparable with t.



(a) Optimization performance with different μ



(b) Primal feasibility of the result with different μ

Figure 2: Comparison of Central Path method with different initial value of μ

4.2 What advantages/disadvantages are there in taking correction steps?

The correction steps are enhancements to the basic primal-dual method. They are introduced to correct any invisibility or to improve the optimality of the current solutions by taking into account the errors in the primal and dual feasibility conditions.

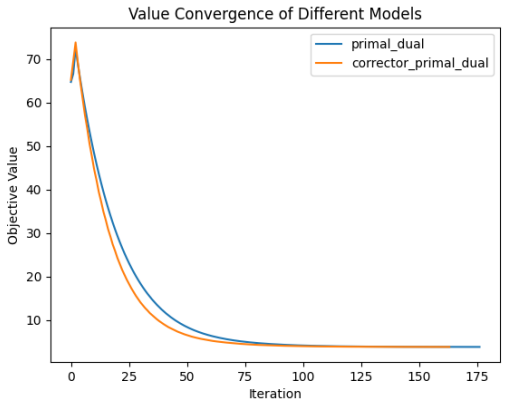
Optimality corrections are aimed at improving the objective values of the solutions. This is typically achieved by moving the solutions in the direction that reduces the duality gap, which is the difference between the primal and dual objective values.

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x^{cor} \\ \Delta \lambda^{cor} \\ \Delta s^{cor} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -\Delta X^{aff} \Delta s^{aff} e \end{bmatrix}$$

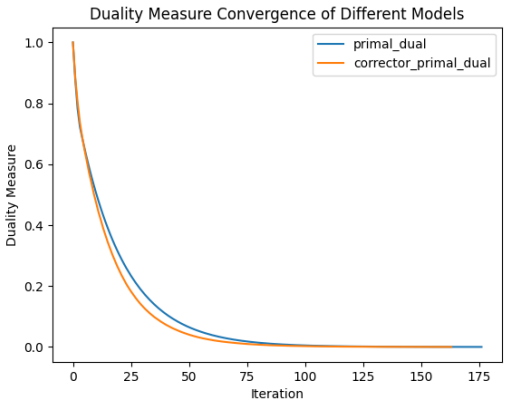
Then we can combine the correction step with the original direction to get a better step.

$$(x_i + \Delta x_i^{aff})(s_i + \Delta s_i^{aff}) = x_i s_i + x_i \Delta s_i^{aff} + s_i \Delta x_i^{aff} + \Delta x_i^{aff} \Delta s_i^{aff} = \Delta x_i^{aff} \Delta s_i^{aff}$$

We compared the basic primal-dual method with the version with correction steps on their convergence speed on object function value as well as its duality gap, the results are shown in 3(a) and 3(b)



(a) Value Convergence Comparison



(b) Duality Measurement Comparison

Figure 3: Comparison between basic primal-dual and primal-dual with corrector step. Version with corrector step give a better estimation of decent direction, it converge faster than the basic version on both the object function value and the duality measurement.

Advantages

1. Enhanced Convergence: Correction steps can significantly accelerate the convergence of an optimization algorithm towards an optimal solution. By adjusting the search direction or step size based on the current state of the solution, these steps can help avoid inefficient paths that slow down convergence.

2. Improved Numerical Stability: Numerical issues can arise during optimization due to factors like ill-conditioned problems or rounding errors. Correction steps can mitigate these problems by adjusting the algorithm's path to maintain numerical stability, ensuring that the iterations produce meaningful progress.

3. Maintaining Feasibility: In constrained optimization problems, maintaining the feasibility of solutions is crucial. Correction steps can help adjust solutions that are close to violating constraints, ensuring that the algorithm remains within the feasible region of the problem space.

Disadvantages

1. Increased Computational Complexity: Implementing correction steps can add computational overhead to each iteration. This is because additional calculations are required to determine when and how to apply these steps, which can increase the overall time to convergence, especially for large-scale problems.

2. Parameter Tuning: Many correction steps involve parameters that need to be carefully tuned to the specific problem being solved. Finding the right parameters can be challenging and time-consuming, requiring extensive experimentation or experience with similar problems. In our cases, we have to balance the step size of the correction steps with the normal optimization steps, ensuring that each adjustment is conducive to progressing towards the solution without causing instability or undue oscillation in the search path. (Details can be found around line 73 of the correction-primal-dual.py in our codebase)

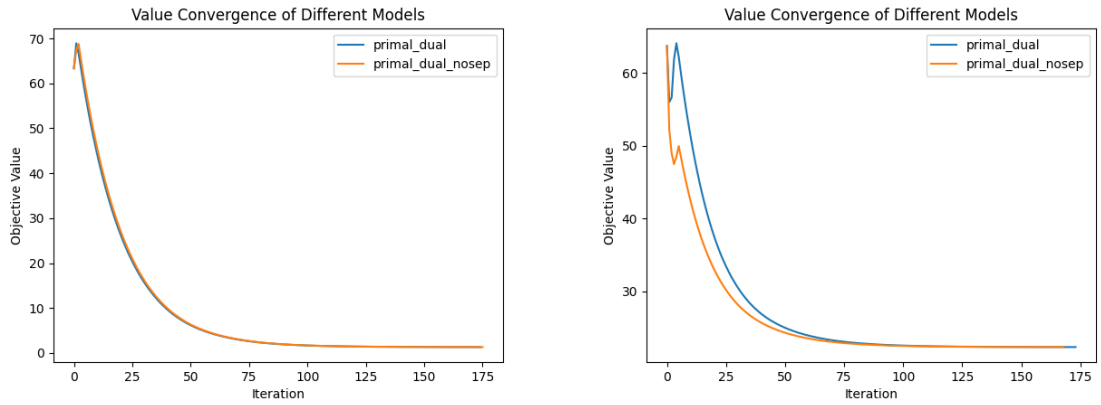
In conclusion, the decision to use correction steps in optimization algorithms involves balancing the potential for improved performance against the added complexity and computational demands. In many cases, the benefits of faster convergence, improved stability, and feasibility maintenance justify the additional overhead. However, the effectiveness of correction steps must be evaluated on a case-by-case basis, in some cases, it's possible for it performs worse than the basic primal-dual method.

4.3 What advantages/disadvantages are there in taking separate primal and dual steps?

Typically we take separate step for the primal and dual, we affine-scale it to ensure a maximal valid step to reduce the duality measurement as well as keep x and s non-negative, which could be represented as:

$$\alpha_{aff}^{pri} \overset{\text{def}}{=} \min\left(1, \min_{i:\Delta x_i^{aff}<0}\left(-\frac{x_i}{\Delta x_i^{aff}}\right)\right), \quad \alpha_{aff}^{dual} \overset{\text{def}}{=} \min\left(1, \min_{i:\Delta s_i^{aff}<0}\left(-\frac{s_i}{\Delta s_i^{aff}}\right)\right)$$

For the non-separate step, we simply take joint step $\alpha = \min(\alpha_{aff}^{pri}, \alpha_{aff}^{dual})$ for both x and s. We compare their performance in 4



(a) num_var=100, num_eq=20 (b) num_var=100, num_eq=50

Figure 4: Object value convergence comparison of joint vs. separate step strategies

In Fig. 4, We tried out two different methods and found out that their effectiveness depends on the scale of the problem. When there are fewer constraints to meet (20 equal constraints, 100 variables), both methods work almost the same, taking separate step performs slightly better. However, when there are more constraints (50 equal constraints, 100 variables), taking joint step converge faster than the separate one.

Advantages

1. Focused Improvement: By updating primal and dual variables separately, the algorithm can focus on improving specific aspects of the problem at each step. This allows for more targeted adjustments. When the number of constraints is relatively low, as shown in 4(a), the optimization problem is less complex. Separate step strategies might offer slight advantages due to their flexibility and the ability to tailor updates more precisely to the primal or dual sides of the problem. This can lead to a more efficient exploration of the solution space when the constraint landscape is less dense.

2. Flexibility in Strategy: Separate steps provide the flexibility to apply different strategies or step sizes to the primal and dual updates. This can be particularly advantageous when the primal and dual parts of the problem have different characteristics or when one side is closer to optimal than the other.

Disadvantages

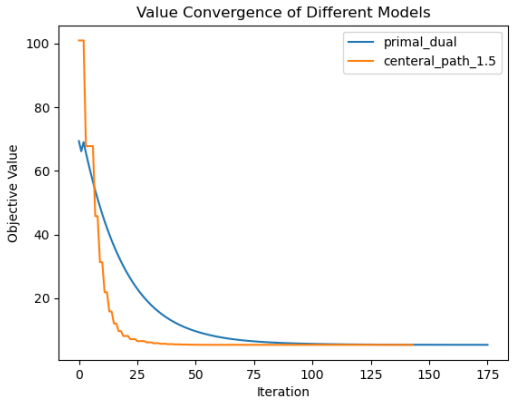
1. Increased Complexity: Managing separate updates for primal and dual variables adds complexity to the algorithm's design and implementation. This can make the optimization method more challenging to understand, develop, and debug, especially for complex problems or for those less familiar with primal-dual methods.

2. Potential for Slower Convergence: As the number of constraints increases as shown in 4(b), the problem becomes more complex, and the inter dependencies between primal and dual variables intensify. In such scenarios, the joint step approach likely achieves faster convergence because it better synchronizes updates to primal and dual variables, maintaining a balance that can navigate the complex constraint landscape more effectively. For the separate step, the updates are not well-coordinated, leading to scenarios where improvements in one part are offset by setbacks in the other, thereby prolonging the path to convergence.

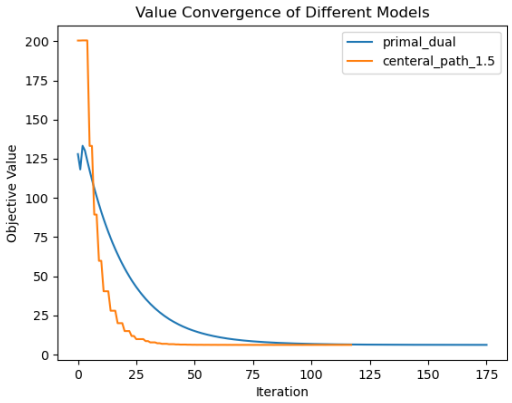
4.4 How well do you expect this method to scale?

To boost our primal-dual solver in order to solve large scale problems, we employed the sparse matrix techniques. We noticed that the jacobian used in the primal-dual method is sparse (at least half of the entries are zeros). This feature gives room for implementing numerical solvers that accelerates computation on sparse matrices. In this project, we used the CSR format provided by Scipy.

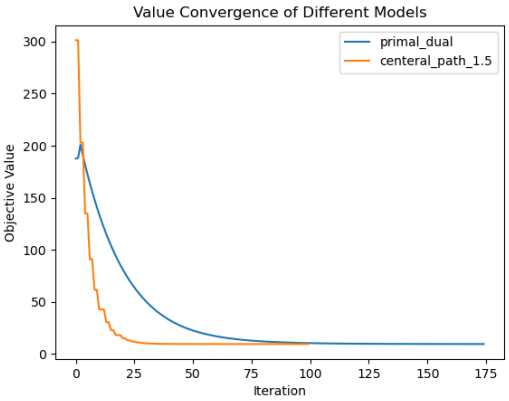
To directly address the question of scalability, it's essential to highlight that our experiments, which spanned problems involving 100 to 400 variables, demonstrate robust evidence of scalability. The number of equations is set to one third of the number of variables to ensure that the constraints are not too sparse. Specifically, the primal-dual method showcased a consistent performance, maintaining efficiency across the tested range. This implies that the method is expected to scale well, even when applied to problems of larger size. This conclusion is drawn from our observations which clearly indicate that there was no significant loss in efficiency as the size of the problems increased within the range of our experiments.



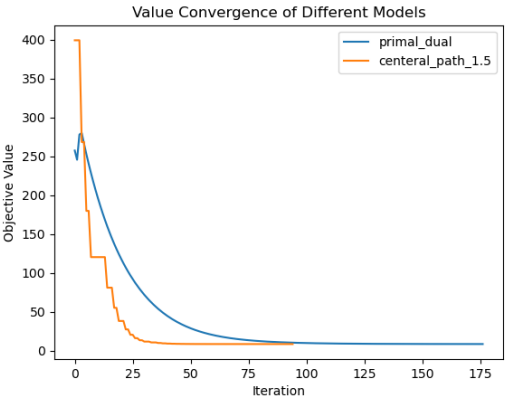
(a) num_var=100, num_eq=33, t_central=4.6s, t_primal=0.45s



(b) num_var=200, num_eq=66, t_central=6.9s, t_primal=1.4s



(c) num_var=300,num_eq=100,t_central=12.6s,t_primal=2.9s

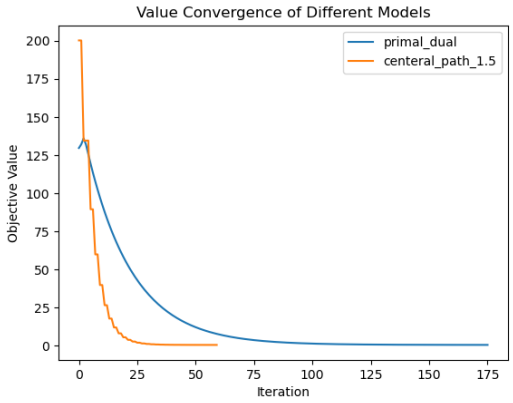


(d) num_var=400,num_eq=133,t_central=16.5s,t_primal=6.1s

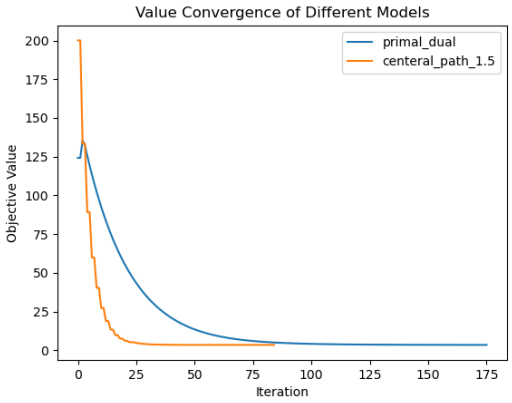
Figure 5: Comparison between Central Path and Primal-Dual performance on large scale problems

4.5 Can you see advantages to very sparse linear constraints?

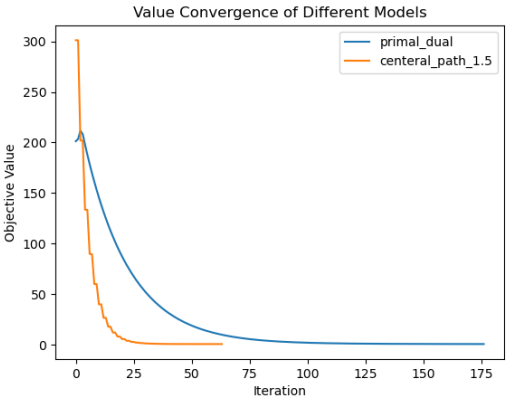
We are primarily interested in comparing the performance of primal-dual method against central path. With the sparse representation, the primal-dual method effectively reduces the scale of computation to match the size of the constraints. As the above graphs show, with sparse constraints, the time needed for the primal-dual method increases only a little with respect to the increase of variables. Compared to section 4.4, we observed that the central path method performs worse if the constraints are sparse. This result shows that the performance of central path depends heavily on the choice of hyper-parameters (t and μ)



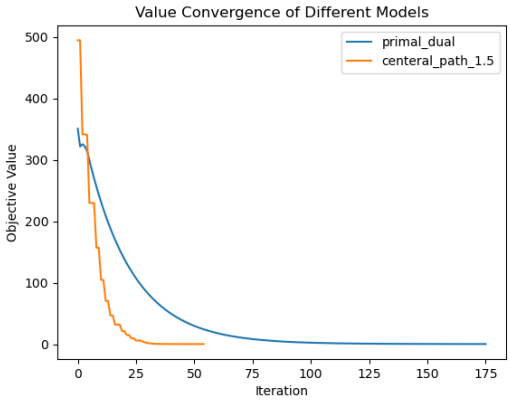
(a) num_var=200, num_eq=10, t_central=5.5s, t_primal=1.0s



(b) num_var=200, num_eq=20, t_central=8.3s, t_primal=1.5s



(c) num_var=300, num_eq=20, t_central=9.5s, t_primal=2.0s



(d) num_var=500, num_eq=10, t_central=30.1s, t_primal=2.8s

Figure 6: Comparison between Central Path and Primal-Dual performance on very sparse problems

5 Conclusion

In this project, we analyzed the performance of interior point methods in a set of different scenarios. We found that in general primal-dual methods outperform central path methods. This is due to their ability to auto adjust the primal-dual gap during optimization, whereas the central path method needs hyper-parameter tuning. Additionally, a number of numerical methods can be implemented on top of the primal-dual method to accelerate the solving process.

References

[1] M. R. Hestenes. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4(5): 303–320, 1969. doi: 10.1007/BF00927673.

[2] H. W. Kuhn and A. W. Tucker. Nonlinear programming proceedings of the 2nd berkeley symposium on mathematical statistics and probability. In *Proceedings of the 2nd Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, Berkeley, 1951. University of California Press.

[3] Dong C. Liu and Jorge Nocedal. On the limited memory method for large scale optimization. *Mathematical Programming B*, 45(3):503–528, 1989. doi: 10.1007/BF01589116.