Krylov Solvers for Interior Point Methods with Applications in Radiation Therapy and Support Vector Machines

Felix Liu[1,2][0000-0001-6865-9379], Albin Fredriksson[2], and Stefano Markidis[1]

[1] KTH Royal Institute of Technology, Stockholm, Sweden
felixliu@kth.se
[2] RaySearch Laboratories, Stockholm, Sweden

Abstract. Interior point methods are widely used for different types of mathematical optimization problems. Many implementations of interior point methods in use today rely on direct linear solvers to solve systems of equations in each iteration. The need to solve ever larger optimization problems more efficiently and the rise of hardware accelerators for general purpose computing has led to a large interest in using iterative linear solvers instead, with the major issue being inevitable ill-conditioning of the linear systems arising as the optimization progresses. We investigate the use of Krylov solvers for interior point methods in solving optimization problems from radiation therapy and support vector machines. We implement a prototype interior point method using a so called doubly augmented formulation of the Karush-Kuhn-Tucker linear system of equations, originally proposed by Forsgren and Gill, and evaluate its performance on real optimization problems from radiation therapy and support vector machines. Crucially, our implementation uses a preconditioned conjugate gradient method with Jacobi preconditioning internally. Our measurements of the conditioning of the linear systems indicate that the Jacobi preconditioner improves the conditioning of the systems to a degree that they can be solved iteratively, but there is room for further improvement in that regard. Furthermore, profiling of our prototype code shows that it is suitable for GPU acceleration, which may further improve its performance in practice. Overall, our results indicate that our method can find solutions of acceptable accuracy in reasonable time, even with a simple Jacobi preconditioner.

Keywords: Interior point method · Krylov solver · Radiation therapy · Support Vector Machines

1 Introduction

Mathematical optimization is used in many areas of science and industry, with applications in fields like precision medicine, operations research and many others. In this paper, we focus on the solution of quadratic programs (QP), which are optimization problems with a quadratic objective function and linear constraints, using an interior point method (IPM). These arise in many applications naturally, for instance in training support vector machine classifiers, but can also be used as part of a sequential quadratic programming solver [3] to solve more general nonlinear optimization problems.

Computationally, IPMs for optimization rely on Newton's method to find search directions. This involves the solution of a large, often sparse and structured linear system of equations, which we will refer to henceforth as the Karush-Kuhn-Tucker (KKT) system, at each iteration. Traditionally, this system is often solved using direct linear solvers, such as LDL^T-factorization for indefinite matrices or Cholesky factorization for positive definite methods, but a topic of interest for much research in the field is the use of iterative linear solvers [23] instead. Indeed, the move to iterative linear algebra for interior point methods has been identified by some authors as a key step in enabling interior point methods to handle very large optimization problems [15].

Another advantage of iterative algorithms for solving linear systems is their suitability for modern computing hardware, such as GPUs. With their rising dominance in High-Performance Computing (HPC), extracting the maximum performance from modern computing hardware all but requires the use of some type of accelerator. Direct linear solvers can suffer performance wise on these types of hardware for a variety of reasons, e.g., unstructured memory accesses, which has been seen in previous studies [25] in the context of interior point methods. The major challenge of using iterative solvers lies in a structured form of ill-conditioning that inevitably exists in the linear systems, which can be severely detrimental to the convergence of the linear solver. Still, given potential performance gains from the use of significantly more powerful computing resources, we believe the trade-off between numerical stability and parallel performance is worthy of further investigation.

One type of problems we consider arise from treatment planning for radiation therapy, which loosely speaking is the process of optimizing treatment plans (treatment machine parameters) for each individual patient case to deliver as accurate a dose as possible to the tumor volume. This inverse problem is often solved by formulating it as a constrained optimization problem, for which finding a solution can be both computationally expensive and present an important bottleneck in the clinical workflow. Computational speed and efficiency is thus of crucial importance, and in the ideal case the optimization would be performed in real time, with the patient present at the clinic. To demonstrate the applicability of our proposed method to other problems as well, we also consider problems from the training of support vector machine classifiers [19], an important method from classical machine learning. All in all, we are interested in studying and evaluating the potential of using IPMs with iterative linear solvers as an avenue to enable us to utilize accelerators and powerful computing resources for solving these problems more efficiently.

In this paper, we propose a complete IPM solver prototype based on the work by Forsgren and Gill in [11], where a special formulation of the KKT systems from interior point methods is considered, which guarantees that the KKT system is positive-definite throughout the optimization for convex problems. Our contribution a complete IPM solver prototype using the doubly augmented formulation and iterative linear solvers, which is capable of solving real world optimization problems. We demonstrate its effectiveness for the solution of quadratic optimization problems arising from real world applications in both radiation therapy treatment planning as well as support vector machines.

2 Background

2.1 Interior Point Methods

We are concerned with the solution of convex, continuous quadratic programs using interior point methods. The following section introduces the relevant aspects for our proposed method. Readers interested in a more thorough overview of interior point methods for optimization are referred to e.g. [26,13]. In general, we will be dealing with a problem on the form:

$$\begin{aligned} \text{min.} \quad & \frac{1}{2}x^T Hx + p^T x \\ \text{s.t.} \quad & l \leq Ax \leq u \\ & Cx = b \end{aligned} \tag{1}$$

where H is the (positive definite) Hessian of the objective function, $p \in \mathbb{R}^n$ are the linear coefficients of the objective, A, C are the Jacobians of the (linear) inequality- and equality constraints, respectively. In general, we allow components of the constraints to be unbounded, but we do not account for this explicitly to simplify the exposition. Inequality constraints are often treated by the introduction of auxiliary *slack variables*. We convert the problem to having only lower bounds, introduce slack variables s_l, s_u (for lower and upper bounds respectively), use a log-barrier term for the inequality constraints and finally, we use a penalty barrier method [10,13] to handle equality constraints, giving:

$$\begin{aligned} \text{min.} \quad & \frac{1}{2}x^T Hx + p^T x - \\ & - \mu \sum \log((s_l)_i) - \mu \sum \log((s_u)_i) + \\ & + \frac{1}{2\mu}||Cx - b||^2 \\ \text{s.t.} \quad & Ax - s_l - l = 0 \\ & -Ax - s_u + u = 0 \end{aligned} \tag{2}$$

Here μ is the so called barrier parameter. Intuitively, the barrier terms diverge towards $+\infty$ as the boundary of the feasible set is approached, thus encouraging feasibility throughout the iterations. As is common in primal-dual interior point methods, we work with the *perturbed* optimality conditions. These state that an optimal solution to the equality constrained subproblem in (2) satisfies the following:

$$\begin{aligned} r_H &:= Hx + p - A^T\lambda_l + A^T\lambda_u - C^T\lambda_e = 0 \\ r_{A_l} &:= Ax - s_l - l = 0 \\ r_{A_u} &:= -Ax + s_u + u = 0 \\ r_e &:= Cx - b + \mu\lambda_e = 0 \\ r_{c_1} &:= (\lambda_l)_i(s_l)_i - \mu = 0, \quad i = 1, ..., m_l \\ r_{c_2} &:= (\lambda_l)_i(s_u)_i - \mu = 0, \quad i = 1, ..., m_l. \end{aligned} \tag{3}$$

These conditions are very similar to the first order Karush-Kuhn-Tucker conditions for optimality [20, Ch. 12.3], except that the final two conditions are perturbed by μ, and the inclusion of the penalty barrier method for equality constraints. Primal-dual interior point methods generally seek points satisfying the perturbed optimality conditions above using, e.g., Newton's method, while successively decreasing the barrier parameter $\mu \to 0$. As μ approaches 0, we expect our solution to approach a point satisfying the KKT conditions for optimality.

2.2 Optimization Problems in Radiation Therapy and SVMs

The first type of problem we consider are from radiation therapy, and are all exported from the treatment planning system RayStation, developed by the Stockholm-based company RaySearch Laboratories. The problems all arise from *treatment planning* for radiation therapy, where an optimization problem is solved to determine a *treatment plan* for each individual patient. A view of treatment planning from an optimization perspective can be found in e.g. [8,9].

The optimization problems from RayStation are QP-subproblems in a Sequential Quadratic Programming solver used for nonlinear optimization, and have the form:

$$\begin{aligned} \text{min.} \quad & \frac{1}{2}p^T\nabla_{xx}\mathcal{L}(x,\lambda)p + (\nabla f)^T(x)p \\ \text{s.t.} \quad & \nabla g(x)^T p + g(x) \geq 0. \end{aligned} \tag{4}$$

Here, $\mathcal{L}(x,\lambda) = f(x) - g(x)\lambda^T$ is the Lagrangian, and λ are the Lagrange multipliers. In practice, the Hessian of the Lagrangian $\nabla_{xx}\mathcal{L}(x,\lambda)$ can be expensive to form, and it is common to use a quasi-Newton type approximation of the Hessian instead. The SQP solver uses Broyden-Fletcher-Goldfarb-Shanno (BFGS) updates [4] to estimate the Hessian of the non-linear problem, which means that the Hessian for each of our QP-subproblems can be written on matrix form as:

$$H = H_0 + UWU^T, \tag{5}$$

where H_0 is the initial approximation to the Hessian, the dense $n \times k$ matrix U consists of the update vectors on the columns, and W is a diagonal matrix with the scalar update weights on the diagonal. With a suitable line-search method, the updates to the Hessian can be ensured to preserve positive definiteness, thus making the QP-subproblem we need to solve at each SQP iteration convex.

The second type of problem we consider are QP dual problems from support vector machine training for classification. These problems are of the form:

$$\begin{aligned} \text{min.} \quad & \frac{1}{2}\alpha^T H\alpha - \alpha^T e \\ \text{s.t.} \quad & \alpha^T y = 0 \\ & 0 \le \alpha \le c, \end{aligned} \tag{6}$$

where the Hessian H is of the form $H = yy^T Q$, and the entries of Q are $K(x_i, x_j)$, for some "kernel" K used to map the data into a (more) separable space. For our experiments, we use a radial basis function kernel

$$K(x_i, x_j) = \exp(-||x_i - x_j||^2/2\sigma).$$

3 Prototype Interior Point Method

We now describe the key components of our prototype interior point method implementation used in this paper.

3.1 KKT System Formulation

Many optimization problems from real applications include bounds on the (primal) variables themselves. Such bounds can be included by the introduction of appropriate rows in the A matrix. For an efficient formulation, we will consider the bounds on variables separately from general linear constraints. Separating the handling of the variable bounds (i.e. lower and upper bounds on the values of the variables x) and then using Newton's method to solve the perturbed optimality conditions (3) gives a linear system of the form:

$$\begin{pmatrix} H & -A^T & A^T & -I & I & C^T & & & & \\ C & & & & & M & & & & \\ A & & & & & & -I & & & \\ -A & & & & & & & -I & & \\ I & & & & & & & & -I & \\ -I & & & & & & & & & -I \\ & S_{l_A} & & & & & \Lambda_{l_A} & & & \\ & & S_{u_A} & & & & & \Lambda_{u_A} & & \\ & & & S_{l_x} & & & & & \Lambda_{l_x} & \\ & & & & S_{u_x} & & & & & \Lambda_{u_x} \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta\lambda_{l_A} \\ \Delta\lambda_{u_A} \\ \Delta\lambda_{l_x} \\ \Delta\lambda_{u_x} \\ \Delta\lambda_e \\ \Delta s_{l_A} \\ \Delta s_{u_A} \\ \Delta s_{l_x} \\ \Delta s_{u_x} \end{pmatrix} = \begin{pmatrix} -r_H \\ -r_e \\ -r_{l_A} \\ -r_{u_A} \\ -r_{l_x} \\ -r_{u_x} \\ -r_{c_1} \\ -r_{c_2} \\ -r_{c_3} \\ -r_{c_4} \end{pmatrix}, \tag{7}$$

where $\lambda_e, \lambda_{l_A}, \lambda_{u_A}, \lambda_{l_x}, \lambda_{u_x}$ are the Lagrange multipliers for the equality constraints, lower and upper bounds on the (general) linear constraints, and lower and upper bounds for the variable bounds respectively. The slack variables s are subscripted in the same way. The residuals on the RHS are similar to the ones shown in (3). Λ, S, M denote diagonal matrices with the corresponding Lagrange multipliers, slack variables or barrier parameter μ on the diagonal respectively, and e is an appropriately sized column-vector with a value of 1 in all coefficients.

The above system can be reduced in size by block-row elimination. Multiplying the sixth and seventh block row by $\Lambda_{l_A}^{-1}$ and the fifth block row by $\Lambda_{u_A}^{-1}$ and adding them to the second and third rows, followed by multiplying the sixth and seventh block rows by $S_{l_x}^{-1}$ and $-S_{u_x}^{-1}$ respectively and adding to the top row, as well as multiplying the fourth and fifth block rows by $S_{l_x}^{-1}\Lambda_{l_x}$ and $-S_{u_x}^{-1}\Lambda_{u_x}$ and adding to the top row. The final reduced linear system of equations (with the same row operations on the RHS) can be written as:

$$\begin{pmatrix} Q & -B^T \\ B & D \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta\lambda_A \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \end{pmatrix}, \tag{8}$$

where:

$$Q = H + S_{l_x}^{-1}\Lambda_{l_x} + S_{u_x}^{-1}\Lambda_{u_x}, \quad B = \begin{pmatrix} C \\ A \\ -A \end{pmatrix}, \quad D = \begin{pmatrix} M & & \\ & \Lambda_{l_A}^{-1} S_{l_A} & \\ & & \Lambda_{u_A}^{-1} S_{u_A} \end{pmatrix}.$$

$$\Delta\lambda_A = \begin{pmatrix} \Delta\lambda_e \\ \Delta\lambda_{l_A} \\ \Delta\lambda_{u_A} \end{pmatrix}, \quad r_1 = -r_H - S_{l_x}^{-1} r_{c_3} + S_{u_x}^{-1} r_{c_4} - S_{l_x}^{-1}\Lambda_{l_x} r_{l_x} + S_{u_x}^{-1}\Lambda_{u_x} r_{u_x}$$

$$r_2 = \begin{pmatrix} -r_e \\ -r_{A_l} - \Lambda_{l_A}^{-1} r_{c_1} \\ -r_{A_u} - \Lambda_{u_A}^{-1} r_{c_2} \end{pmatrix}$$

At this point the purpose of handling the variable bounds separately becomes clear, since they now only contribute a diagonal term $S_{l_x}^{-1}\Lambda_{l_x} + S_{u_x}^{-1}\Lambda_{u_x}$ in the Hessian block. A challenge with the system (8) is that it becomes inevitably ill-conditioned as the optimization approaches a solution. Intuitively this can be seen by noting that as $\mu \to 0$, some elements of the diagonal block D become very small and some become unbounded, since for active constraints the slack variables tend to zero, while for inactive constraints the Lagrange multipliers do. For more details on this ill-conditioning see, e.g., [27,12]. The system (8) is unsymmetric, and it is common to consider many equivalent but symmetric systems instead. Our implementation uses the *doubly augmented* formulation, proposed in [11]. This formulation can be derived through block-row operations on the system in (8), by multiplying the second block row by $2B^TD^{-1}$ and adding it to the first block row:

$$\begin{pmatrix} Q + 2B^TD^{-1}B & B^T \\ B & D \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta\lambda_A \end{pmatrix} = \begin{pmatrix} r_1 + 2B^TD^{-1}r_2 \\ r_2 \end{pmatrix}. \tag{9}$$

The major advantage is that the matrix is symmetric and positive definite for convex problems [11]. This enables us to use a preconditioned conjugate gradient method to solve the system efficiently. To precondition the system we use Jacobi (diagonal scaling) preconditioning, which is motivated by the ill-conditioning arising primarily from the diagonal D block in the matrix.

3.2 IPM Implementation

We implement a prototype interior point method to assess the performance and accuracy of the method when using iterative linear algebra. The following is a brief description of the design of our implementation. Our implementation is a primal-dual interior point method based on the KKT system formulation in (9). We use a ratio test to determine the maximum step length to take each iteration to maintain the positivity of the slack variables and Lagrange multipliers:

$$\begin{aligned} \alpha_x &= \min\left\{1.0, \gamma\left(\min\left\{-\frac{s_i}{\Delta s_i} : \Delta s_i < 0\right\}\right)\right\} \\ \alpha_\lambda &= \min\left\{1.0, \gamma\left(\min\left\{-\frac{\lambda_i}{\Delta \lambda_i} : \Delta \lambda_i < 0\right\}\right)\right\}. \end{aligned} \tag{10}$$

The scalar $\gamma < 1$ is to ensure strict positivity of the slacks and Lagrange multipliers throughout the optimization, and we use a value of $\gamma = 0.99$ in our implementation. The step lengths from the line search are used to scale the search direction.

Finally, we decrease the barrier parameter μ based value of the residuals (shown in the right-hand side of (7)). Namely, when the 2-norm of the residuals is smaller than the current value of μ, we divide μ by 10 and continue the optimization. The optimizer terminates when the barrier parameter decreases below a tolerance threshold, which by default is set to 10^{-6}.

We summarize the main components of our implementation in Algorithm 1. The use of a Krylov solver for our implementation provides some practical bene-

Algorithm 1 Interior Point Method

```
1:  for $i \leftarrow 1$ to $N$ do
2:      Find search direction by solving (8) using PCG
3:      Line search for $\alpha_x, \alpha_\lambda$ from (10)
4:      $x \leftarrow x + \alpha_x \Delta x$
5:      $\lambda \leftarrow \lambda + \alpha_\lambda \Delta\lambda$
6:      $s \leftarrow s + \alpha_x \Delta s$
7:      Update diagonal $D$ in KKT system
8:      Compute residuals (RHS of (7))
9:      if $||r|| < \mu$ then                    ▷ $r$ is the RHS of (7)
10:         if $\mu < \mu_{tol}$ then
11:             Return solution
12:         end if
13:         $\mu \leftarrow \mu/10$
14:     end if
15: end for
```

fits in how the computation is structured, as it allows us to work in a matrix-free manner. Concretely, we do not explicitly form the $2B^T D^{-1} B$ term in (8) (nor the entirety of the matrix), but always work with the different components separately. This is also especially advantageous for the quasi-Newton structure of the Hessian. as discussed in Section 2.2. Recall that the BFGS-Hessian can be written in matrix form as $H = H_0 + UWU^T$, which is a dense $n \times n$ matrix, where n is the number of variables in the QP. Similarly to before, this matrix is also not explicitly formed in our solver, saving significant computational effort when the number of variables is large.

We have implemented the method described in C++ (using BLAS for many computational kernels), which is also the implementation we use for the experiments conducted in this work.

4 Experimental setup

Table 1: Problem dimensions for the considered QPs

Problem	Vars.	Lin. cons.	Bound cons.
Proton H&N	55770	0	90657
Proton Liver	90657	15	90657
Photon H&N	13425	42273	13425
SVM a1a	1605	1	3210

The radiation therapy optimization problems evaluated in this work are exported directly from the RayStation optimizer. We export the QP subproblems to files directly from RayStation which permits us to use them for our experiments, without relying on RayStation itself. In particular, we consider three problem cases, two cases treated using proton therapy, one for the head-and-neck region and a liver case and one case treated using photons. The dimensions of the corresponding optimization problems are shown in Table 1. In all considered problems, the problems are exported from the later stages of the SQP iterations, which are typically the most challenging.

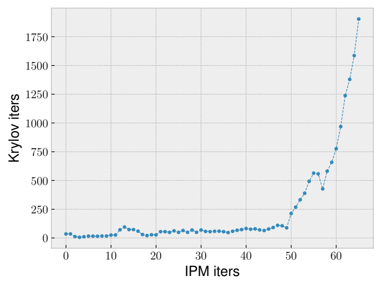
For the SVM training problem, we use the `a1a` problem available from the LIBSVM dataset [5]. We pre-compute the (dense) Hessian H using the radial basis function kernel as described in Section 2.2.

The performance measurements were all carried out on a local workstation equipped with an AMD Ryzen 7900x CPU with 64 GB of DDR5 DRAM. The BLAS library used was OpenBLAS 0.3.21 with OpenMP threading. The measurements of the condition number were run on a node on the Dardel supercomputer in PDC at the KTH Royal Institute of Technology, for memory reasons.

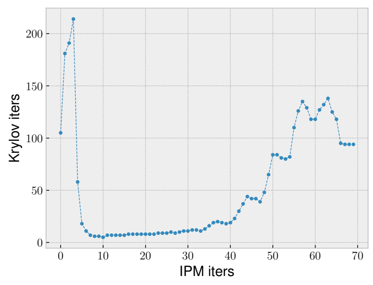
5 Results

In the following section, we present some experimental evaluation of our prototype method in terms of convergence of the conjugate gradient solver and interior point solver itself, as well as the conditioning of the KKT-systems and the computational performance.

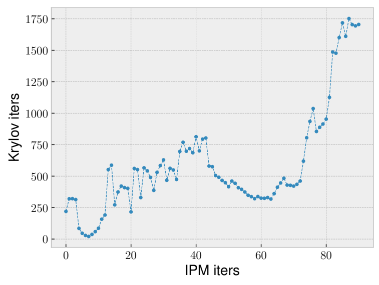
5.1 Krylov Solver Convergence



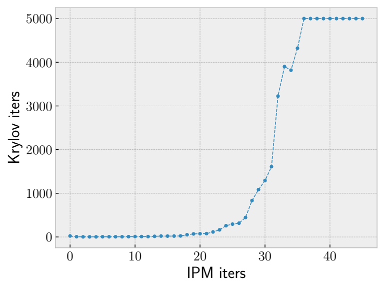
(a) Photon Head and Neck case



(b) Proton Head and Neck case



(c) Proton Liver Case

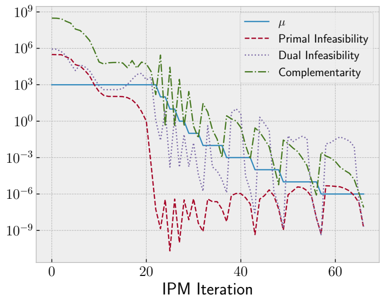


(d) SVM a1a

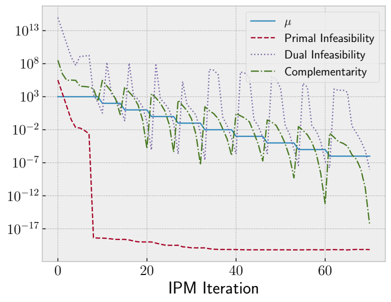
Fig. 1: Number of CG iterations at each IPM iteration for three different test problems.

Fig. 1 shows the number of CG iterations required for the linear solver to converge within each IPM iteration for our test problems. Note that the maximum number of CG iterations was set to 5000 for each of the test cases, with a convergence tolerance of 10^{-7} for the unpreconditioned residual. As a trend, we see that all problems show a sharp increase in the number of CG iterations towards the later IPM iterations, which is consistent with the observation that the ill-conditioning of the systems arises when the barrier parameter μ gets close to zero. The proton head and neck case stands out in that it has a a spike in CG at the beginning of the optimization as well, which is also seen in the estimated condition numbers being large in the beginning in Fig. 3. To note is that the proton head and neck case is the only one considered without general linear constraints (it has only variable bound constraints). For some of the considered cases, especially the proton liver case and SVM cases, the number of CG iterations for convergence is very large, indicating that improved preconditioning is an interesting prospect for future work.

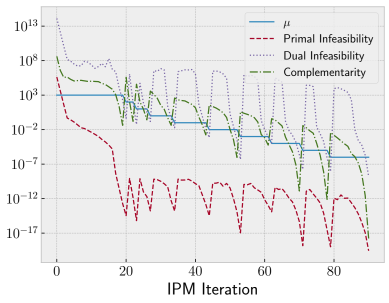
5.2 IPM Solver Convergence



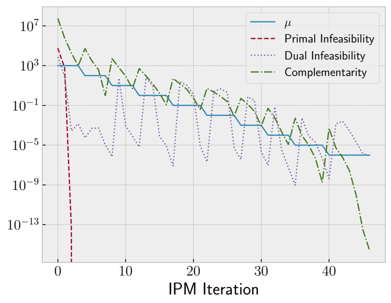
(a) Photon Head and Neck case



(b) Proton Head and Neck case



(c) Proton Liver Case



(d) SVM a1a

Fig. 2: Convergence of our solver in terms of primal, dual and complementarity infeasibility. The value of the barrier parameter μ is shown as well, which is successively decreased as optimization progresses.

Another interesting aspect to consider is the convergence of the interior point method as a whole. In optimization solvers, the convergence is often measured with respect to the *primal*, *dual* and *complementarity* infeasibility (among others). The primal infeasibility is the (Euclidean) norm of $\left(r_{l_A}\ r_{u_A}\ r_{l_x}\ r_{u_x}\right)^T$, the dual infeasibility is the norm of r_H and the complementarity infeasibility the norm of $\left(r_{c_1}\ r_{c_2}\ r_{c_3}\ r_{c_4}\right)^T$. In other words, the primal infeasibility measures the error with respect to satisfying the constraints, the dual infeasibility measures the error in stationarity of the Lagrangian, and the complementarity infeasibility the error with respect to the perturbed complementary slackness condition.

In Fig 2, we show the convergence in terms of the primal, dual and complementarity infeasibility over IPM iterations for our test-problems. From the figures, we see that the convergence towards optimality is far from monotonous, with the spikes in the infeasibility norms coinciding with the points when the barrier parameter μ is decreased in the solver. This could indicate that the update of μ is too aggressive. The reason could be that we use a relatively crude update rule for μ in our prototype implementation, and more sophisticated methods may give better performance. Another interesting observation from the infeasibility plots is that the dual infeasibility exhibits slower convergence compared to the complementarity and primal infeasiblities, especially for the proton cases.

Speculatively, one can observe that the dual infeasibility r_H, appears only once in the right-hand side of (8). The block in the RHS in which the dual infeasibility appears is also contains multiple terms scaled by the inverse of the slack variables, which may be large for *active* constraints. The doubly augmented system (9) introduces an additional term in the top block of the RHS. In view of Krylov methods as algorithms seeking least-norm solutions in a given Krylov subspace for a set of linear system of equations, this may give a partial explanation why the dual infeasibility lags behind in our case, as the contribution to the RHS in the linear system from the corresponding term can be relatively small.

5.3 Numerical Stability and Conditioning of KKT System

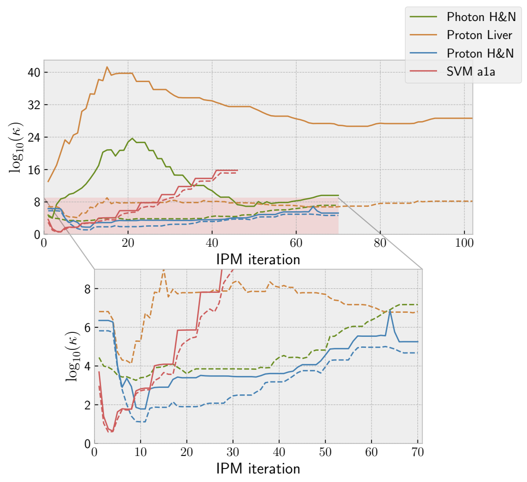


Fig. 3: Condition number estimates using Matlab's `condest` throughout the optimization iterations. The bottom plot is zoomed in on the red region (with the Photon H&N line removed for clarity). The dashed lines are condition numbers after Jacobi preconditioning.

As discussed previously, one of the main challenges in using iterative linear solvers for interior point methods is the structured ill-conditioning in the linear systems. To study how this conditioning affects our problem, we evaluate how the condition number κ of the doubly augmented KKT system (9) that we solve in each iteration changes throughout the optimization. The condition numbers of the resulting matrices are estimated using Matlab's `condest` function, which we modify slightly by using Cholesky instead of LU-factorization internally, since our matrices are symmetric positive definite. `condest` gives an estimate of the condition number in the $L1$-norm and is based on an algorithm proposed by Higham in [16].

Fig. 3 shows the results of the condition number analysis. The solid lines show the un-preconditioned condition numbers, with the dashed lines showing the condition numbers with Jacobi preconditioning. The un-preconditioned KKT systems for the Photon H&N and Proton Liver cases show extreme ill-conditioning, especially in the middle of the optimization, with estimated condition numbers up to the order 10^{41} for the Proton Liver case and 10^{23} for the Photon H&N case. While the accuracy of Matlab's estimation using `condest` at such extreme ill-conditioning may be questioned, suffice it to say that the un-preconditioned matrices are close to singular. However, we see that the Jacobi preconditioning does manage to improve the conditioning of those matrices significantly, reducing the condition number to around 10^8 (or less) for both cases.

5.4 Performance Analysis

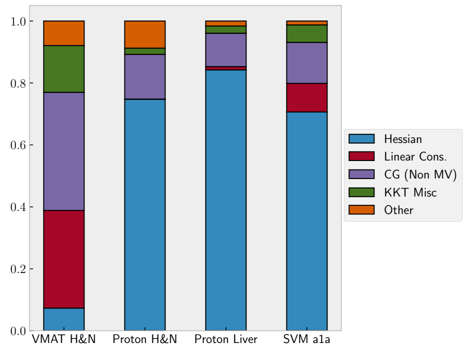


Fig. 4: Relative (normalized) run-time spent in different computational kernels for our solver.

Fig. 4 shows the relative time spent in different parts of the code for the three test problems considered in this work. The absolute run-times for the different parts is given in Table 2. The parts we included in the profiling are: Matrix-vector multiplication with the quasi-Newton Hessian, matrix-vector multiplication with

Table 2: Total run times in seconds spent in different parts of the solver for the different problems. CG (Not MV) is the time spent in the conjugate gradients solver excluding matrix-vector products. The relative times in different parts are also visualized in Fig. 4.

Case	Total	Hessian	Lin. Cons.	CG (Not MV)	KKT Misc.	Other
Photon H&N	13.6	0.997	4.30	5.20	2.06	1.08
Proton H&N	6.28	4.70	0.002	0.907	0.127	0.551
Proton Liver	324	222	62.8	28.4	6.13	4.29
SVM a1a	2.78	1.96	0.256	0.367	0.157	0.035

the constraint matrix, remaining time in the CG solver (excluding matrix vector multiplication) remaining time spent in the matrix-vector multiplication with the doubly augmented KKT system in (9), and finally the "Other" category comprising the remaining time spent in other parts of the solver.

From Fig. 4, we see that for both proton cases and the SVM problem, the solver spends the majority of the time in computing (dense) matrix-vector products with the Hessian, while for the photon radiation therapy problem, a significant amount of time is spent in the CG solver itself, as well as for multiplication with the linear constraints. Overall, we believe the performance analysis shows that there is potential for improved performance when moving to GPU, especially for the proton radiation therapy problems and SVM problems.

6 Related Work

The topic of iterative linear solvers in interior point methods has attracted much research, which we briefly summarize in the following section. Preconditioners for KKT systems in interior point method have been studied extensively previously, for instance in [2,17,14,21,7,28] among many others. A general overview of HPC in the space of optimization and optimization software can be found in [18]. The topic of parallel computing in optimization and operations research in general has also been surveyed previously [24].

Practical studies where iterative linear solvers are used for different kinds of optimization problems can be found in [22], where a type of hybrid direct-iterative solution method is evaluated on very large problems in optimal power flow. In the context of interior point methods for linear programming (LP), preconditioned Krylov methods were studied in e.g.[7,6].

7 Conclusions and Future Work

In this work, we have presented our prototype interior point method for quadratic programming that uses an iterative linear solver for the KKT systems arising in each iteration. We demonstrate that the method can solve real optimization problems from radiation therapy to acceptable levels of accuracy and within reasonable time. From analyzing the performance of our implementation using tracing and profiling, we believe that our method is suitable for GPU acceleration, which we will investigate further in future work. Overall, we believe that interior point methods using Krylov solvers give a promising path forward for GPU accelerated interior point methods, which hold great promise for e.g. computational efficiency in treatment plan optimization for radiation therapy.

There are many interesting questions and problems remaining for future research. Among those is the porting of the code to be able to run on GPU accelerators and looking at improved preconditioners for the KKT systems. One concrete possiblity for improved preconditioning would be to consider a method similar to the one proposed in [1], based on low-ranks updates of a factorized Schur-complement. For some of the problems considered in this paper with only a few linear constraints, and thus a correspondingly small Schur complement, explicit re-factorization of the Schur complement may be so cheap that the low-rank update scheme is not required at all. Finally, another interesting possibility for future work is to investigate the suitability of the proposed method for optimization problems from other domains.

8 Acknowledgment

The computations were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) at PDC, partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

References

1. Bellavia, S., De Simone, V., di Serafino, D., Morini, B.: Updating constraint preconditioners for kkt systems in quadratic programming via low-rank corrections. SIAM Journal on Optimization **25**(3), 1787–1808 (2015)
2. Bergamaschi, L., Gondzio, J., Zilli, G.: Preconditioning indefinite systems in interior point methods for optimization. Computational Optimization and Applications **28**, 149–171 (2004)
3. Boggs, P.T., Tolle, J.W.: Sequential quadratic programming. Acta numerica **4**, 1–51 (1995)
4. Broyden, C.G.: The convergence of a class of double-rank minimization algorithms 1. general considerations. IMA Journal of Applied Mathematics **6**(1), 76–90 (1970)
5. Chang, C.C., Lin, C.J.: Libsvm: a library for support vector machines. ACM transactions on intelligent systems and technology (TIST) **2**(3), 1–27 (2011)
6. Chowdhury, A., Dexter, G., London, P., Avron, H., Drineas, P.: Faster randomized interior point methods for tall/wide linear programs. Journal of Machine Learning Research **23**(336), 1–48 (2022)
7. Cui, Y., Morikuni, K., Tsuchiya, T., Hayami, K.: Implementation of interior-point methods for lp based on krylov subspace iterative solvers with inner-iteration preconditioning. Computational optimization and applications **74**, 143–176 (2019)
8. Ehrgott, M., Güler, Ç., Hamacher, H.W., Shao, L.: Mathematical optimization in intensity modulated radiation therapy. Annals of Operations Research **175**(1), 309–365 (2010)

9. Engberg, L.: Automated radiation therapy treatment planning by increased accuracy of optimization tools. Ph.D. thesis, KTH Royal Institute of Technology (2018)
10. Fiacco, A.V., McCormick, G.P.: Nonlinear programming: sequential unconstrained minimization techniques. SIAM (1990)
11. Forsgren, A., Gill, P.E., Griffin, J.D.: Iterative solution of augmented systems arising in interior methods. SIAM Journal on Optimization **18**(2), 666–690 (2007)
12. Forsgren, A., Gill, P.E., Shinnerl, J.R.: Stability of symmetric ill-conditioned systems arising in interior methods for constrained optimization. SIAM Journal on Matrix Analysis and Applications **17**(1), 187–211 (1996)
13. Forsgren, A., Gill, P.E., Wright, M.H.: Interior methods for nonlinear optimization. SIAM review **44**(4), 525–597 (2002)
14. Gill, P.E., Murray, W., Ponceleón, D.B., Saunders, M.A.: Preconditioners for indefinite systems arising in optimization. SIAM journal on matrix analysis and applications **13**(1), 292–311 (1992)
15. Gondzio, J.: Interior point methods 25 years later. European Journal of Operational Research **218**(3), 587–601 (2012)
16. Higham, N.J.: Fortran codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation. ACM Transactions on Mathematical Software (TOMS) **14**(4), 381–396 (1988)
17. Karim, S., Solomonik, E.: Efficient preconditioners for interior point methods via a new schur complement-based strategy. SIAM Journal on Matrix Analysis and Applications **43**(4), 1680–1711 (2022)
18. Liu, F., Fredriksson, A., Markidis, S.: A survey of hpc algorithms and frameworks for large-scale gradient-based nonlinear optimization. The Journal of Supercomputing **78**(16), 17513–17542 (2022)
19. Noble, W.S.: What is a support vector machine? Nature biotechnology **24**(12), 1565–1567 (2006)
20. Nocedal, J., Wright, S.: Numerical Optimization. Springer Series in Operations Research and Financial Engineering, Springer New York, New York, NY, 2nd ed. 2006. edn. (2006)
21. Rees, T., Greif, C.: A preconditioner for linear systems arising from interior point optimization methods. SIAM Journal on Scientific Computing **29**(5), 1992–2007 (2007)
22. Regev, S., Chiang, N.Y., Darve, E., Petra, C.G., Saunders, M.A., Świrydowicz, K., Peleš, S.: Hykkt: a hybrid direct-iterative method for solving kkt linear systems. Optimization Methods and Software pp. 1–24 (2022)
23. Saad, Y.: Iterative methods for sparse linear systems. SIAM (2003)
24. Schryen, G.: Parallel computational optimization in operations research: A new integrative framework, literature review and research directions. European Journal of Operational Research **287**(1), 1–18 (2020)
25. Świrydowicz, K., Darve, E., Jones, W., Maack, J., Regev, S., Saunders, M.A., Thomas, S.J., Peleš, S.: Linear solvers for power grid optimization problems: A review of gpu-accelerated linear solvers. Parallel Computing **111**, 102870 (2022)
26. Wright, M.H.: Interior methods for constrained optimization. Acta numerica **1**, 341–407 (1992)
27. Wright, M.H.: Ill-conditioning and computational error in interior methods for nonlinear programming. SIAM Journal on Optimization **9**(1), 84–111 (1998)
28. Zilli, G., Bergamaschi, L.: Block preconditioners for linear systems in interior point methods for convex constrained optimization. ANNALI DELL'UNIVERSITA'DI FERRARA **68**(2), 337–368 (2022)