ORIGINAL PAPER

Computing tighter bounds on the *n*-queens constant via Newton's method

Parth Nobel[1] · **Akshay Agrawal[1]** · **Stephen Boyd[1]**

Abstract

In recent work Simkin shows that bounds on an exponent occurring in the famous *n*-queens problem can be evaluated by solving convex optimization problems, allowing him to find bounds far tighter than previously known. In this note we use Simkin's formulation, a sharper bound developed by Knuth, and a Newton method that scales to large problem instances, to find even sharper bounds.

Keywords *n*-queens constant · Combinatorial optimization · Convex optimization · Newton's method

1 Introduction

Let $\mathcal{Q}(n)$ denote the number of ways that *n* queens can be arranged on an $n \times n$ chessboard in such a way that none is threatening another, *i.e.*, no two queens are in the same row, column, or diagonal. Recent work by Simkin [1] has shown that

$$\lim_{n \to \infty} \frac{\mathcal{Q}(n)^{1/n}}{n} = e^{-\alpha},$$

where α is a constant, that we refer to as the *n*-queens constant, characterized as the optimal value of an infinite dimensional convex optimization problem. For background on the problem and previously derived bounds on $Q(n)$, see [2].

In [1], Simkin establishes that $\alpha \in [1.94, 1.9449]$, a strong tightening of the best previously known bounds $\alpha \in [1.58, 3]$ [3, 4]. His method finds lower and upper bounds by solving two convex optimization problems. Knuth later formulated another convex optimization problem which also gives an upper bound on α [5].

✉ Parth Nobel
  ptnobel@stanford.edu

1   Department of Electrical Engineering, Stanford University, 350 Jane Stanford Way, Stanford, CA 94305, USA

In this note we solve the convex optimization problems associated with Simkin's lower bound and Knuth's upper bound, using a version of Newton's method that scales to large problem instances, to establish that

$$\alpha \in [1.944000752, 1.944001082].$$

This agrees with previous conjectures that $\alpha \approx 1.944$ [6]. In terms of the gap, *i.e.*, difference of known upper and lower bounds, Simkin improved it from the previous value around 1.4 to around 5×10^{-3}, and we have improved that to around 3.3×10^{-7}.

Simkin's numerical lower bound is found as a lower bound on the optimal value of a convex optimization problem whose optimal value is a lower bound on the n-queens constant. This problem is parameterized by n, the size of the chessboard used to interpret the problem. We let L_n denote the optimal objective value of this problem. In [5], Knuth introduces a convex optimization problem whose optimal value is an upper bound on the n-queens constant. It is also parameterized by n, and we let U_n denote its optimal value. Simkin's numerical upper bound is found by solving a related convex optimization problem which upper bounds Knuth's problem.

Simkin's numerical bounds are a lower bound on L_{17} and an upper bound on U_{12}, obtained by approximately solving these two problems. These problems involve a few hundred variables and constraints. In contrast, we use Newton's method to solve the problems, which has two advantages. First, we solve the problem to high accuracy, so almost nothing is lost when we move from an approximate solution to a lower or upper bound on the optimal value. Second, our method scales to much larger n, which gives us tighter bounds on the n-queens constant.

Our numerical bounds are L_{2048} and U_{1024}, obtained by solving the two problems to high accuracy. The lower bound problem contains almost 17 million variables and over 12000 constraints; the upper bound problem contains over 4 million variables and over 14000 constraints. In this note we explain how a version of Newton's method can be used to solve such large problems. (The numbers $n = 1024$ and $n = 2048$ are chosen as the largest powers of two that we can evaluate on the desktop computer we used to carry out the computations).

2 The convex problems

The bounds L_n and U_n are the optimal values of convex optimization problems of the form

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b, \end{array} \tag{1}$$

where $x \in \mathbf{R}^p$ is the variable, $A \in \mathbf{R}^{q \times p}$, $b \in \mathbf{R}^q$ specify the constraints, and the objective function $f : \mathbf{R}_{++}^p \to \mathbf{R}$ is smooth and strictly convex. ($\mathbf{R}_{++}$ denotes the set of positive numbers). These problems are feasible, and so have a unique solution.

In these optimization problems, f, A, and b are parametrized by n, but to lighten the notation we suppress this dependence in our description of the method. Full descriptions of f, A, and b for the lower bound and upper bound problems are given in the appendix. Here, we summarize some of their attributes.

Lower bound problem. For the lower bound problem, we have $p = 4n^2 + 4n$ variables and $q = 6n - 1$ constraints. The objective f is separable, *i.e.*, a sum of functions of x_i, so its Hessian $\nabla^2 f(x)$ is diagonal. The constraint coefficient matrix A is full rank and sparse, with at most $4n$ nonzero entries in each row, and at most 4 nonzero entries in each column. The entries of A are all 0 or 1.

Upper bound problem. For the upper bound problem, we have $p = 4n^2 + 8n - 4$ variables and $q = 14n - 6$ constraints. The objective f is block separable, a sum of functions of pairs of variables, where the pairs are disjoint, so its Hessian $\nabla^2 f(x)$ is block diagonal, with 1×1 and 2×2 blocks. Here too A is full rank and sparse, with at most 4 nonzero entries per column, and at most $2n + 1$ nonzero entries per row. Its entries are all either 0 or 1 or $2n$.

3 Infeasible start Newton method

In this section we summarize the infeasible start Newton method described in [7, §10.3.2] (which also contains a convergence proof), and explain how to compute the search directions in a scalable way. We also discuss how to compute appropriate bounds on optimal values of the problems.

3.1 Optimality condition and residuals

The necessary and sufficient optimality conditions for (1) are

$$\nabla f(x) + A^T v = 0, \qquad Ax - b = 0,$$

where $v \in \mathbf{R}^q$ is a dual variable or Lagrange multiplier. For $x \in \mathbf{R}_{++}^p$ and $v \in \mathbf{R}^q$ we define the dual and primal residuals as

$$r_d(x, v) = \nabla f(x) + A^T v, \qquad r_p(x, v) = Ax - b, \tag{2}$$

and the (primal-dual) residual $r(x, v) = (r_d(x, v), r_p(x, v))$. Thus the optimality condition can be expressed as $r(x, v) = 0$.

3.2 Infeasible start Newton method

The method is iterative, with iterates denoted as $(x^{(k)}, v^{(k)})$, where k is the iteration number. The iterates will satisfy $x^{(k)} \in \mathbf{R}_{++}^p$, so the residual $r^{(k)} = r(x^{(k)}, v^{(k)})$ is defined (and will converge to zero as $k \to \infty$). We initialize our algorithm with $x^{(0)} \in \mathbf{R}_{++}^p$, which need not satisfy $Ax^{(0)} = b$.

Newton step For the kth iterate the Newton step $(\Delta x^{(k)}, \Delta v^{(k)}) \in \mathbf{R}^p \times \mathbf{R}^q$ is the solution of the linear equations

$$r(x^{(k)}, v^{(k)}) + Dr(x^{(k)}, v^{(k)})(\Delta x^{(k)}, \Delta v^{(k)}) = 0,$$

where Dr is the derivative or Jacobian of the residual. (We will show later that these equations always have a unique solution.) The lefthand side is the first order Taylor approximation of $r(x^{(k)} + \Delta x^{(k)}, v^{(k)} + \Delta v^{(k)})$, so if the Newton step is added to the current iterate, we obtain primal and dual variables for which the Taylor approximation is zero.

We write the equations defining the Newton step as

$$\begin{bmatrix} \nabla^2 f(x^{(k)}) & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x^{(k)} \\ \Delta v^{(k)} \end{bmatrix} = - \begin{bmatrix} r_d(x^{(k)}, v^{(k)}) \\ r_p(x^{(k)}, v^{(k)}) \end{bmatrix}. \tag{3}$$

The coefficient matrix is invertible, since its top left block is invertible and its bottom left block is wide and full rank; see, *e.g.*, [8, §16.2] or [7, §10.1.1].

From the top block equations in (3) we have

$$\Delta x^{(k)} = -\nabla^2 f(x^{(k)})^{-1} \left(r_d(x^{(k)}, v^{(k)}) + A^T \Delta v^{(k)} \right). \tag{4}$$

Substituting this into the bottom block of equations we obtain the set of equations

$$\left(A\nabla^2 f(x^{(k)})^{-1} A^T \right) \Delta v^{(k)} = r_p(x^{(k)}, v^{(k)}) - A\nabla^2 f(x^{(k)})^{-1} r_d(x^{(k)}, v^{(k)}), \tag{5}$$

with positive definite coefficient matrix $S = A\nabla^2 f(x^{(k)})^{-1} A^T$. To find the Newton step, we first solve the set of equations (5) to obtain $\Delta v^{(k)}$, and then evaluate $\Delta x^{(k)}$ using (4).

Line search and update. The next iterate has the form

$$x^{(k+1)} = x^{(k)} + t^{(k)} \Delta x^{(k)}, \qquad v^{(k+1)} = v^{(k)} + t^{(k)} \Delta v^{(k)},$$

where $t^{(k)}$ is a positive step length. Choosing $t^{(k)}$ is referred to as a line search. Our line search is one specifically for the infeasible start Newton method, described in [7, §10.3.2]; for more general discussion of line search methods, see, *e.g.*, [9, Chap. 3].

To find $t^{(k)}$ we first find $\tilde{t} = \min(0.95 t_{\max}, 1)$, where

$$t_{\max} = \min \left\{ \frac{x_i^{(k)}}{-\Delta x_i^{(k)}} \;\middle|\; \Delta x_i^{(k)} < 0 \right\}$$

is the largest possible step for which $x^{(k)} + t\Delta x^{(k)} \in \mathbf{R}_+^p$. We take $t^{(k)} = \beta^\ell \tilde{t}$, where $\beta \in (0, 1)$ is a parameter and ℓ is the smallest positive integer for which

$$\|r(x^{(k)} + \beta^\ell \tilde{t} \Delta x^{(k)}, v^{(k)} + \beta^\ell \tilde{t} \Delta v^{(k)})\|_2 \leq (1 - \alpha\beta^\ell \tilde{t})\|r(x^{(k)}, v^{(k)})\|_2$$

holds, where $\alpha \in (0, 1/2)$ is a parameter. (It can be shown that such an integer exists [7, §10.3.1].) If $\|r(x^{(k+1)}, v^{(k+1)})\|_2 < \epsilon$, we terminate, where ϵ is a positive tolerance.

We use the common line search parameter values $\alpha = 0.01$ and $\beta = 0.9$, and the tolerance $\epsilon = 10^{-9}$, which is far smaller than would be needed in any engineering or statistics application.

Efficient computation. The computational effort in each step is, predominantly, solving the linear equations (5). Since we intend to use our algorithm on problem instances where forming and storing the $q \times q$ matrix $S = A\nabla^2 f(x^{(k)})^{-1}A^T$ is not practical, we use an indirect iterative method to solve these equations [10]. There are many such methods, mostly based on Krylov subspaces, such as conjugate gradients [11]. The particular method we use is MINRES [12]. Like other indirect methods, it requires only a method to evaluate the mapping $y \mapsto Sy$ for a vector y. We evaluate this mapping as

$$Sy = A\big(\nabla^2 f(x^{(k)})^{-1}\big(A^T y\big)\big),$$

i.e., successive multiplications by A^T, $\nabla^2 f(x^{(k)})^{-1}$, and A, without forming or storing the matrix S. (Simply storing S for the specific problems we will solve would require many terabytes of memory.)

3.3 Bounds on optimal values

To bound the *n*-queens constant α, we need a lower bound on L_n and an upper bound on U_n. Newton's method is able to solve the lower bound and upper bound problems to high accuracy, so the optimal values we compute could simply be rounded down or up to obtain these bounds. Here we discuss ways to more carefully compute these bounds on the optimal values. To find an upper bound on U_n, it suffices to find a feasible x and evaluate the objective at that point.

Lower bounds on L_n. We follow Simkin and use standard Lagrangian duality to find a lower bound on L_n. The dual function of (1) is

$$h(v) = v^T b - f^*(A^T v),$$

where f^* is the conjugate function of f [7, §3.3]. For any v, $h(v)$ is a lower bound on the optimal value of the problem (1). For the lower bound problem we can explicitly find f^* as

$$f^*(y) = \sum_{i=1}^{p} \exp\big(y_i - 1\big) - 4\log n - 2\log 2 - 3$$

via considering the structure of f as given in §A.2 and a straightforward application of results given in [7, §3.3.1]. To obtain a lower bound on L_n, we solve the problem using Newton's method and then evaluate $h(v)$ for the v found. (Since we solve these problems to high accuracy, the lower bound on L_n obtained is very close to the upper bound on L_n found, which is $f(x)$.)

Rational approximation. The bounds described above are found using floating point computations. To make the upper bound fully precise, we find a rational approximation of x that is *exactly* feasible and evaluate the objective, carefully using an upper bound on the (transcendental) objective function. For the lower bound, we find a rational approximation of v and evaluate a lower bound on the dual function. We have not taken these steps, because our floating point solutions are so accurate that it would have a negligible effect on our final numerical bounds.

4 Results

Lower bound. We computed L_{2048} by solving a problem with $p = 16785408$ variables and $q = 12288$ constraints. This required 21 iterations, with a total time of around 517 seconds on a M1 Mac Mini. We obtained the lower bound

$$1.944000752019729 = L_{2048}.$$

Upper bound. We computed U_{1024} by solving a problem with $p = 4202492$ variables and $q = 14332$ constraints. To speed up finding the solution, we started Newton's method from (x, v) that solve the approximate upper bound problem (§A.4), which has a simpler objective and so was faster to compute. This required 6 iterations to solve the approximate problem, and a further 7 iterations to solve the exact problem. The total time was around 56 seconds on a M1 Mac Mini. We obtained the upper bound

$$1.9440010813092217 = U_{1024}.$$

Our code is available at https://github.com/cvxgrp/n-queens.

Appendix

A Details of convex problems

In this appendix we give the details of the lower and upper bound problems, as well as an approximate upper bound problem. We define the variables in their natural notation, leaving it to the reader to re-arrange these into a single vector variable x. In a similar way, we describe the linear constraints in their natural notation, leaving it to the reader to translate these into $Ax = b$.

A.1 Common notation

In this section we describe variables and notation that are shared by the lower and upper bound problems.

Chessboard triangle variables. In both problems, the variable x consists of 4 $n \times n$ matrices N, E, S, W, and some additional slack variables. The i, jth entry in N, E, S, W is interpreted as a value associated with the North, East, South, or West triangle, respectively, formed by dividing each square of an $n \times n$ chessboard into 4 right triangles. We index these matrices starting at 0, diverging from the notation used in [1, 5] which use indexing that begins at 1. Figure 1 shows the $n = 8$ case.

Diagonal sum operators. We introduce operators $\mathcal{D}_k : \mathbf{R}^{n \times n} \to \mathbf{R}$ and $\mathcal{A}_k : \mathbf{R}^{n \times n} \to \mathbf{R}$ defined for $k \in \{-n, -n+1, -n+2, \ldots, -1, 0, 1, \ldots, n-2, n-1, n\}$, where $\mathcal{D}_k(Z)$ is the sum of the kth diagonal of Z, and $\mathcal{A}_k(Z)$ is the sum of the kth anti-diagonal of Z. For example $\mathcal{D}_0 Z = \sum_{i=0}^{n-1} Z_{ii} = \mathbf{Tr} Z$, $\mathcal{D}_1 Z = \sum_{i=1}^{n-1} Z_{i,i-1}$, and $\mathcal{A}_{-1} Z = \sum_{i=1}^{n-1} Z_{n-i,i}$. Note that $\mathcal{D}_n Z = \mathcal{D}_{-n} Z = 0$. These are illustrated in Fig. 1:
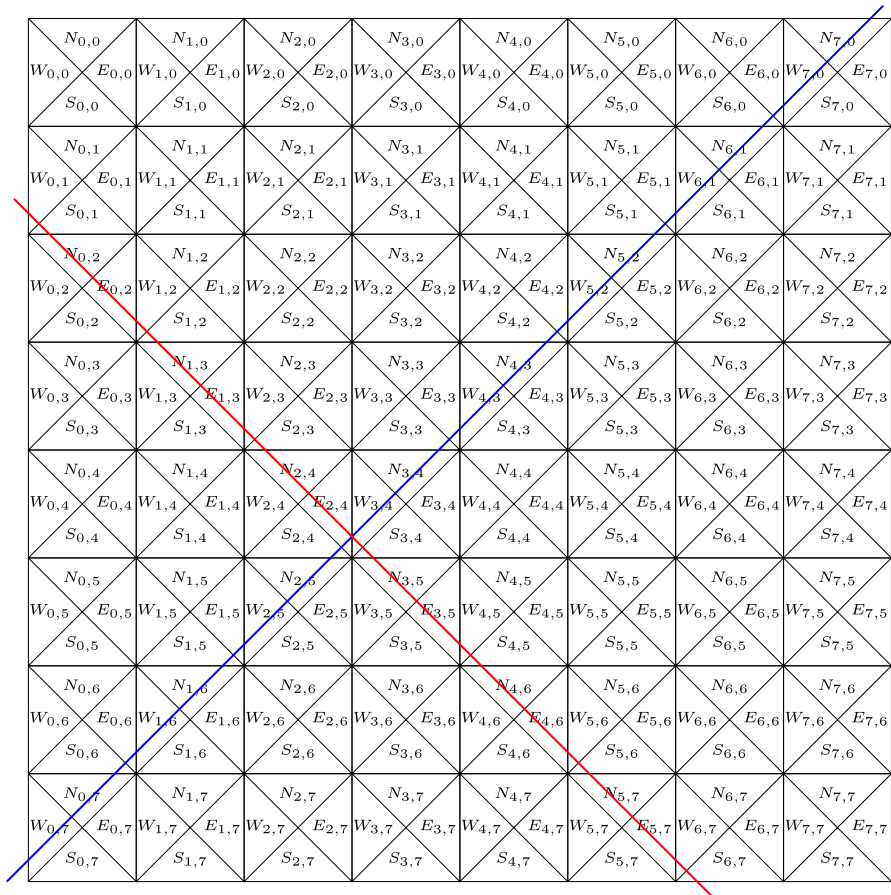
Fig. 1 A chessboard with all of its triangles labeled. This chessboard is used to interpret the $n = 8$ problem. The red line represents one of the diagonals of the chessboard. The blue line represents one of the anti-diagonals of the chessboard

$\mathcal{D}_{-2}N$ is the sum of the entries in the North triangles the red line passes through, and $\mathcal{A}_1 E$ is the sum of the entries in the East triangles the blue line passes through.

Negative entropy. Following [5], we define the function $g : \mathbf{R}_+ \to \mathbf{R}$ as $g(x) = x \log x$ for $x > 0$, and $g(0) = 0$. (This is the negative entropy function [7, p.72].)

A.2 Lower bound problem

This problem formulation is taken from [1, Claim 6.3], except that Simkin maximizes a concave function and we minimize its negative, a convex function.

Slack variables We introduce the following slack variables,

$$d_k = \frac{1}{n} - \mathcal{D}_k(S + W) - \mathcal{D}_{k+1}(N + E), \quad k \in \{-n, -n + 1, \ldots, n - 1\},$$

and

$$a_k = \frac{1}{n} - \mathcal{A}_k(S + E) - \mathcal{A}_{k+1}(N + W), \quad k \in \{-n, -n + 1, \ldots, n - 1\}.$$

The quantities d_k and a_k are the sums along the diagonals and anti-diagonals, respectively, of the chessboard. In Fig. 1, d_{-1} includes contributions from all triangles the red line passes through and a_0 includes contributions from triangles the blue line passes through.

These equations form $4n$ entries in A, b.

Objective. The objective function is

$$\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \left(g(N_{i,j}) + g(E_{i,j}) + g(S_{i,j}) + g(W_{i,j}) \right)$$

$$+ \sum_{k=-n}^{n-1} \left(g(d_k) + g(a_k) \right) + 4 \log n + 2 \log 2 + 3.$$

Constraints. Simkin also introduces the constraints

$$\sum_{j=0}^{n-1} N_{i,j} + E_{i,j} + S_{i,j} + W_{i,j} = \frac{1}{n}, \quad i \in \{0, 1, \ldots, n - 1\},$$

and

$$\sum_{i=0}^{n-1} N_{i,j} + E_{i,j} + S_{i,j} + W_{i,j} = \frac{1}{n}, \quad j \in \{0, 1, \ldots, n - 1\}.$$

These constraints are linearly dependent, with co-rank one, so we delete the first constraint with $i = 0$ to obtain a total of $2n - 1$ constraints that we include in A, b.

Properties. This problem has n^2 entries in each of N, E, S, W and $2n$ entries in each of d, a. Accordingly, the total number of variables is $p = 4n^2 + 4n$. We have $4n$ constraints affecting the slack variables, and $2n - 1$ constraints affecting only N, E, S, W for a total of $q = 6n - 1$ constraints.

The objective is a sum of the negative entropy of individual optimization variables, making it separable and strictly convex.

The constraint with the most variables are the row and column constraints, which involve $4n$ variables. Each triangle is in at most 1 column constraint, 1 row constraint, 1 diagonal constraint, and 1 anti-diagonal constraint. Therefore, each column of A can have at most 4 entries.

A.3 Upper bound problem

We use Knuth's formulation of the Xqueenon problem [5], except that he maximizes a concave function and we minimize its negative, a convex function.

Slack variables. We introduce the slack variables

$$d_k^{SW} = 1 - \frac{1}{2n}\mathcal{D}_k(S + W), \quad k \in \{-n + 1, -n + 2, \dots, n - 1\},$$

$$d_k^{NE} = 1 - \frac{1}{2n}\mathcal{D}_k(N + E), \quad k \in \{-n + 1, -n + 2, \dots, n - 1\},$$

$$a_k^{SE} = 1 - \frac{1}{2n}\mathcal{A}_k(S + E), \quad k \in \{-n + 1, -n + 2, \dots, n - 1\},$$

and

$$a_k^{NW} = 1 - \frac{1}{2n}\mathcal{A}_k(N + W), \quad k \in \{-n + 1, -n + 2, \dots, n - 1\}.$$

These equations form $8n - 4$ entries in A, b.

Objective. For ease of notation, let

$$d_{-n}^{SW} = d_n^{NE} = a_{-n}^{SE} = a_n^{NW} = 1.$$

Our objective function is

$$3 + L_0(N, E, S, W) + L_-(d^{SW}, d^{NE}) + L_+(a^{SE}, a^{NW}),$$

where

$$L_0(N, E, S, W) = \frac{1}{4n^2} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1}$$
$$\big(g(N_{i,j}) + g(E_{i,j}) + g(S_{i,j}) + g(W_{i,j})\big),$$
$$L_-(d^{SW}, d^{NE}) = \frac{1}{n} \sum_{k=-n+1}^{n} \int_0^1 g\big((1 - y)d_{k-1}^{SW} + yd_k^{NE}\big)\, dy,$$

and

$$L_+(a^{SE}, a^{NW}) = \frac{1}{n} \sum_{k=-n+1}^{n} \int_0^1 g\big((1 - y)a_{k-1}^{SE} + ya_k^{NW}\big)\, dy.$$

Using a symbolic solver, we were able to generate closed-form expressions for the integrals and their partial derivatives [13].

In order to make the matrix block-diagonal, d_k^{SW} and d_k^{NE} must be interleaved in x. Similar interleaving applies to a_k^{SE} and a_k^{NW}.

Constraints. In addition to the $8n - 4$ equations involving the slack variables, Knuth requires the following conditions on N and S,

$$\sum_{j=0}^{n-1} N_{i,j} = n, \quad i \in \{0, 1, \ldots, n-1\},$$

$$\sum_{j=0}^{n-1} S_{i,j} = n, \quad i \in \{0, 1, \ldots, n-1\},$$

and

$$\sum_{i=0}^{n-1} N_{i,j} + S_{i,j} = 2n, \quad j \in \{0, 1, \ldots, n-1\}.$$

As any of these equations are linearly dependent on all the others, we choose to eliminate the first column constraint on N.

On E and W, Knuth requires

$$\sum_{i=0}^{n-1} E_{i,j} = n, \quad j \in \{0, 1, \ldots, n-1\},$$

$$\sum_{i=0}^{n-1} W_{i,j} = n, \quad j \in \{0, 1, \ldots, n-1\},$$

and

$$\sum_{j=0}^{n-1} E_{i,j} + W_{i,j} = 2n, \quad i \in \{0, 1, \ldots, n-1\}.$$

As with N and S, one of these equations is linearly dependent, and we choose to eliminate the first row constraint on E.

Properties. This problem has n^2 entries in each of N, E, S, W and $2n - 1$ entries in each of $d_k^{SW}, d_k^{NE}, a_k^{SE}, a_k^{NW}$. This forms a total of $p = 4n^2 + 8n - 4$ variables. We also have the $8n - 4$ constraints involving the slack variables and $6n - 2$ of the other constraints for a total of $q = 14n - 6$ constraints. The objective is block separable as each variable appears in only one term of the objective function and no term has more than two variables. The rows of A with the most entries are the entries along the diagonal and anti-diagonal, which contain $2n$ entries of N, E, S, W and 1 slack variable. Each column of A has at most 4 non-zero entries: 1 from its row constraint, 1 from its column constraints, 1 from its diagonal term, and 1 from its anti-diagonal term. Columns associated with slack variables have one non-zero entry.

A.4 Approximate upper bound problem

In the initial phase of computing U_n we solve a problem with a diagonal Hessian that approximates the upper bound problem. We do this by applying Jensen's inequality to the integrals in the diagonal and anti-diagonal terms of the objective.

After applying this approximation, the integral terms of the objective function become

$$g\left(\frac{1}{2}d_{k-1}^{SW} + \frac{1}{2}d_k^{NE}\right),$$

and

$$g\left(\frac{1}{2}a_{k-1}^{SE} + \frac{1}{2}a_k^{NW}\right).$$

We introduce new slack variables $d_k = \frac{1}{2}d_{k-1}^{SW} + \frac{1}{2}d_k^{NE}$ and $a_k = \frac{1}{2}a_{k-1}^{SE} + \frac{1}{2}a_k^{NW}$ and then replace the integral terms with $g(d_k)$ and $g(a_k)$ appropriately.

All other constraints and terms of the objective function are the same.

References

1. Simkin, M.: The number of *n*-queens configurations (2021). arXiv:2107.13460 [math.CO]
2. Bell, J., Stevens, B.: A survey of known results and research areas for *n*-queens. Discret. Math. **309**(1), 1–31 (2009)
3. Luria, Z.: New bounds on the number of *n*-queens configurations (2017). arXiv:1705.05225 [math.CO]
4. Luria, Z., Simkin, M.: A lower bound for the *n*-queens problem (2021). arXiv:2105.11431 [math.CO]
5. Knuth, D.: Xqueens and Xqueenons (2021). https://cs.stanford.edu/~knuth/papers/Xqueens-and-Xqueenons.pdf
6. Zhang, C., Ma, J.: Counting solutions for the *n*-queens and Latin-square problems by Monte Carlo simulations. Phys. Rev. E. Stat. Nonlinear Soft Matter. Phys. **79**, 016703 (2009)
7. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
8. Boyd, S., Vandenberghe, L.: Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares. Cambridge University Press, Cambridge (2018)
9. Nocedal, J., Wright, S.: Numerical Optimization. Springer, Berlin (2006)
10. Saad, Y.: Iterative Methods for Sparse Linear Systems. SIAM, Philadelphia (2003)
11. Hestenes, M., Stiefel, E.: Methods of conjugate gradients for solving linear systems. J. Natl. Bureau Stand. **49**(6), 409–4361 (1952)
12. Paige, C., Saunders, M.: Solution of sparse indefinite systems of linear equations. SIAM J. Numer. Anal. **12**(4), 617–629 (1975)
13. The Sage Developers. SageMath, the Sage Mathematics Software System (Version 9.4), 2021. https://www.sagemath.org