Quantum key distribution rates from non-symmetric conic optimization

Andrés González Lorente, Pablo V. Parellada, Miguel Castillo-Celeita, and Mateus Araújo

Departamento de Física Teórica, Atómica y Óptica, Universidad de Valladolid, 47011 Valladolid, Spain
5th March 2025

Computing key rates in quantum key distribution (QKD) numerically is essential to unlock more powerful protocols, that use more sophisticated measurement bases or quantum systems of higher dimension. It is a difficult optimization problem, that depends on minimizing a convex non-linear function: the (quantum) relative entropy. Standard conic optimization techniques have for a long time been unable to handle the relative entropy cone, as it is a non-symmetric cone, and the standard algorithms can only handle symmetric ones. Recently, however, a practical algorithm has been discovered for optimizing over non-symmetric cones, including the relative entropy. Here we adapt this algorithm to the problem of computation of key rates, obtaining an efficient technique for lower bounding them. In comparison to previous techniques it has the advantages of flexibility, ease of use, and above all performance.

1 Introduction

Secret key rates in QKD are usually calculated analytically [1–4]. This is only tractable for simple protocols with highly symmetric measurement bases, such as BB84 [5], the six-state protocol [6], or their generalizations to mutually unbiased bases (MUBs) in higher dimensions [7, 8]. Recently there has been intense interest in numerical techniques for the computation of key rates in order to unlock more sophisticated protocols, that use more parameters, arbitrary measurement bases, and higher dimensions.

The best existing techniques are the Gauss-Newton technique from [9], and the Gauss-Radau technique from [10]. The former consists of an implementation of a bespoke Gauss-Newton solver for the key rate problem, and the latter consists of reducing the problem to a convergent hierarchy of semidefinite programs. Both allow for the computation of optimal key rates for arbitrary protocols, but suffer from several disadvantages. The main one, common to both techniques, is low performance, which limits their applicability to protocols with low dimension. Specific problems of the Gauss-Newton technique are that it doesn't use standard conic optimization techniques, which makes it challenging to implement and inflexible. For example, it cannot handle protocols that use anything other than equality constraints. A specific problem of the Gauss-Radau technique is that for any fixed level of the hierarchy it only delivers an approximation of the key rate, and the cost of achieving a given precision can be prohibitive.

Here we introduce a new technique that is several times faster than both techniques, while also solving their specific issues. Unlike the Gauss-Newton technique, it uses standard conic optimization, making it easy to implement and combine with other kinds of constraints that might be required by the protocols. Unlike the Gauss-Radau technique, it doesn't rely on reformulating the problem as a sequence of semidefinite programs, but instead solves it directly.

We use the framework from [11] to formulate the key rate problem as a convex optimization problem, the minimization of a (quantum) relative entropy over a convex domain. Using the relative entropy cone, which is a non-symmetric cone, it becomes a conic optimization problem. Although for a long time algorithms for optimizing over non-symmetric cones have been known [12–14], they were hardly practical, as they required for instance a tractable barrier function for

the dual cone. This changed with the discovery of Skajaa and Ye's algorithm [15, 16], and with it solving the key rate problem becomes in principle possible. There is, however, a technical difficulty: this formulation of the key rate problem is often not strictly feasible, and such problems cannot be reliably solved with conic optimization methods. The standard technique to deal with this, called facial reduction [17], does apply, but the resulting reduced problem is no longer a minimization of the relative entropy. For this reason we introduce a new convex cone to deal directly with the reduced problem.

We implemented our new cone in the programming language Julia [18] as an extension to the solver Hypatia [19], which implements an improved version of the Skajaa and Ye algorithm [20]. This solver is interfaced by the modeller JuMP [21], making it easy to use, and takes advantage of Julia's flexible type system, allowing the user to solve the optimization problems using double, double-double, quadruple, or arbitrary precision.

The paper is organized as follows. In Section 2 we reformulate the key rate problem as a conic problem and introduce the QKD cone. In Section 3 we propose a barrier function and obtain the derivatives necessary to implement the cone. In Section 4 we show how to formulate some QKD protocols to compute the key rates with our technique. In Section 5 we benchmark our technique against the Gauss-Newton and Gauss-Radau techniques.

2 QKD rate as a conic problem

In quantum key distribution the asymptotic key rate is given by the Devetak-Winter rate [22]:

$$K \geq H(A|E) - H(A|B), \tag{1}$$

where $H(A|E)$ is the conditional von Neumann entropy between Alice and Eve, and $H(A|B)$ the conditional von Neumann entropy between Alice and Bob. Since $H(A|B)$ is completely determined by the measured data, the challenge is to compute $H(A|E)$, which has to be minimized over all quantum states compatible with the measured statistics. Following [11], it can be expressed in terms of the relative entropy as

$$H(A|E)_{\rho_{ABE}} = D(\mathcal{G}(\rho_{AB}) \| \mathcal{Z}(\mathcal{G}(\rho_{AB}))), \tag{2}$$

where $D(\cdot\|\cdot)$ is the relative entropy, $\mathcal{G}$ is a CP map that takes ρ_{AB} to an extended quantum state that includes the secret key as a subsystem, and $\mathcal{Z}$ is a CPTP map that decoheres the key subsystem. $\mathcal{Z}$ is necessarily of the form $\mathcal{Z}(\rho) = \sum_i \Pi_i \rho \Pi_i$ for some projectors Π_i that sum to identity. The optimization problem is thus

$$\begin{aligned} \min_{\rho} \;& D(\mathcal{G}(\rho) \| \mathcal{Z}(\mathcal{G}(\rho))) \\ \text{s.t.} \;& \rho \succeq 0, \quad \operatorname{tr}(\rho) = 1, \\ & \operatorname{tr}(E_k \rho) = p_k \;\forall k \end{aligned} \tag{3}$$

where E_k are the POVMs encoding the QKD protocol, and p_k are the estimated probabilities. This formulation makes the computation of key rates a convex optimization problem. To make it a *conic* optimization problem, however, we need to write it in terms of the relative entropy convex cone, namely

$$\mathcal{K}_{\text{RE}} = \operatorname{cl}\{(h, \rho, \sigma) \in \mathbb{R} \times \mathbb{H}^n \times \mathbb{H}^n;\; \rho \succ 0,\; \sigma \succ 0,\; h \geq D(\rho\|\sigma)\}, \tag{4}$$

where $\mathbb{H}^n$ denotes the space of $n \times n$ complex Hermitian matrices. It becomes then

$$\begin{aligned} \min_{h,\rho} \;& h \\ \text{s.t.} \;& \rho \succeq 0, \quad \operatorname{tr}(\rho) = 1, \\ & \operatorname{tr}(E_k \rho) = p_k \;\forall k, \\ & \Big(h, \mathcal{G}(\rho), \mathcal{Z}(\mathcal{G}(\rho))\Big) \in \mathcal{K}_{\text{RE}} \end{aligned} \tag{5}$$

In principle it can now be solved by using Skajaa and Ye's algorithm, but there's a difficulty: as with any interior-point algorithm, the problem needs to be strictly feasible for the solution to be found reliably and efficiently. In other words, there must exist a point (h, ρ) that satisfies the equality constraints of the conic problem and is in the interior of all cones we use. This can fail here in two places: the equality constraints $\operatorname{tr}(E_k\rho) = p_k$ might be only satisfiable by a quantum state with null eigenvalues, and the map $\mathcal{G}$ might take a positive definite quantum state to one with null eigenvalues. The map $\mathcal{Z}$, on the other hand, preserves positive definiteness[1] [9].

We note that this same difficulty affects other techniques [9–11]: in [11] they perturb the singular matrices with identity to make them full rank, whereas in [9, 10] they reformulate the problem in terms of the support of the relevant matrices, a procedure known as facial reduction [17]. Facial reduction is more reliable, elegant, and results in a smaller problem than perturbing with identity, so we adopt this approach here as well.

We would thus like to perform facial reduction along the lines of [9]: first find an isometry V that encodes the support of the feasible ρ, such that $\rho = V\sigma V^\dagger$ for full-rank σ. Then replace the equality constraints with $\operatorname{tr}(V^\dagger E_k V\sigma) =: \operatorname{tr}(F_k\sigma) = p_k$, dropping the redundant ones. Finally, find maps $\widehat{\mathcal{G}}$ and $\widehat{\mathcal{Z}}$ such that $\widehat{\mathcal{G}}(\sigma)$ and $\widehat{\mathcal{Z}}(\sigma)$ are the restrictions of the maps $\sigma \mapsto \mathcal{G}(V\sigma V^\dagger)$ and $\sigma \mapsto \mathcal{Z}(\mathcal{G}(V\sigma V^\dagger))$ to the support of their ranges.

While that's easy to do, the resulting problem is no longer an optimization over the relative entropy; in particular $D(\widehat{\mathcal{G}}(\sigma)\|\widehat{\mathcal{Z}}(\sigma)) \neq D(\mathcal{G}(\rho)\|\mathcal{Z}(\mathcal{G}(\rho)))$. Instead, to obtain the reformulated problem we use the following identity [23]:

$$D(\mathcal{G}(\rho)\|\mathcal{Z}(\mathcal{G}(\rho))) = -H(\mathcal{G}(\rho)) + H(\mathcal{Z}(\mathcal{G}(\rho))), \tag{6}$$

where $H(\rho) = -\operatorname{tr}(\rho\log\rho)$ is the von Neumann entropy, together with the simple observation that $H(\mathcal{G}(\rho)) = H(\widehat{\mathcal{G}}(\sigma))$ and $H(\mathcal{Z}(\mathcal{G}(\rho))) = H(\widehat{\mathcal{Z}}(\sigma))$. This motivates the definition of a new convex cone:

$$\mathcal{K}_{\mathrm{QKD}}^{\widehat{\mathcal{G}},\widehat{\mathcal{Z}}} = \{(h,\sigma) \in \mathbb{R}\times\mathbb{H}^n;\ \sigma \succeq 0,\ h \geq -H(\widehat{\mathcal{G}}(\sigma)) + H(\widehat{\mathcal{Z}}(\sigma))\}. \tag{7}$$

We emphasize that $\widehat{\mathcal{G}}$ and $\widehat{\mathcal{Z}}$ must come from the reduction of $\mathcal{G}$ and $\mathcal{Z}$; if we let them be arbitrary CP maps $\mathcal{K}_{\mathrm{QKD}}^{\widehat{\mathcal{G}},\widehat{\mathcal{Z}}}$ is not even a convex cone in general.

With the cone in hand, we can finally state the reduced problem:

$$\begin{aligned} &\min_{h,\sigma}\ h \\ \text{s.t.}\quad &\operatorname{tr}(\sigma) = 1, \quad \operatorname{tr}(F_k\sigma) = p_k\ \forall k, \\ &(h,\sigma) \in \mathcal{K}_{\mathrm{QKD}}^{\widehat{\mathcal{G}},\widehat{\mathcal{Z}}} \end{aligned} \tag{8}$$

Note that the constraint $\rho \succeq 0$ has been dropped; although it's equivalent to $\sigma \succeq 0$, the latter is redundant with the definition of $\mathcal{K}_{\mathrm{QKD}}^{\widehat{\mathcal{G}},\widehat{\mathcal{Z}}}$.

We emphasize that even in the case where no facial reduction is necessary, and $\widehat{\mathcal{G}} = \mathcal{G}$ and $\widehat{\mathcal{Z}} = \mathcal{Z}\circ\mathcal{G}$, it is still advantageous to use the QKD cone (7) instead of the relative entropy cone (4) for four reasons: (i) the constraint $\rho \succeq 0$ can not be dropped in the relative entropy cone unless $\mathcal{G}$ has a positive inverse, (ii) the QKD cone has one less matrix variable to optimize over, (iii) the derivatives of the barrier function of the QKD cone, that we introduce below, are less computationally demanding, (iv) as the QKD cone explicitly depends on $\widehat{\mathcal{G}}$ and $\widehat{\mathcal{Z}}$, it can exploit their structure; specifically, we use the fact that $\widehat{\mathcal{Z}}$ necessarily has a block diagonal structure to drastically speed up the computation of several derivatives involving it.

3 Implementation of the QKD cone

In order to successfully optimize over a given convex cone $\mathcal{K}$, both using Skajaa and Ye's algorithm or traditional methods for symmetric cones, it's essential to provide a logarithmically homogeneous

[1] To see that, assume for the sake of contradiction that $|\psi\rangle$ is a null eigenvector of $\mathcal{Z}(\rho)$ for some positive definite ρ. Then $\langle\psi|\mathcal{Z}(\rho)|\psi\rangle = \sum_i \langle\psi|\Pi_i\rho\Pi_i|\psi\rangle = 0$. Since ρ is positive definite, $\langle\psi|\Pi_i\rho\Pi_i|\psi\rangle > 0$ for all nonzero $\Pi_i|\psi\rangle$. Therefore, to satisfy the equation we must have $\Pi_i|\psi\rangle = 0\ \forall i$. Summing over i and using the fact that $\sum_i \Pi_i = \mathbb{1}$, we arrive at $|\psi\rangle = 0$, contradiction.

self-concordant barrier (LHSCB) function for it. It is defined as a function $f : \mathrm{int}(\mathcal{K}) \to \mathbb{R}$ such that $f(x_i) \to \infty$ along every sequence converging to the boundary of $\mathcal{K}$, is three times continuously differentiable, strictly convex, and

$$f(\tau x) = f(x) - \nu \log \tau \quad \forall x \in \mathrm{int}(\mathcal{K}), \forall \tau > 0, \tag{9}$$

$$|\nabla^3 f(x)[\xi, \xi, \xi]| \leq 2(\nabla^2 f(x)[\xi, \xi])^{3/2} \quad \forall x \in \mathrm{int}(\mathcal{K}), \forall \xi, \tag{10}$$

where $\nabla^2 f(x)[\xi, \xi]$ and $\nabla^3 f(x)[\xi, \xi, \xi]$ denote the Hessian and third-order derivative at point x in direction ξ. For an introduction to LHSCB functions and the matrix derivatives see [24].

While most of these properties are straightforward to verify, self-concordance (10) is not, and has only recently been proven for the following barrier function for the relative entropy cone (4) [25]:

$$f(h, \rho, \sigma) = -\log(h - D(\rho\|\sigma)) - \mathrm{logdet}(\rho) - \mathrm{logdet}(\sigma). \tag{11}$$

We propose as the barrier function for the QKD cone (7) its obvious analogue:

$$f(h, \rho) = -\log(h + H(\widehat{\mathcal{G}}(\rho)) - H(\widehat{\mathcal{Z}}(\rho))) - \mathrm{logdet}(\rho). \tag{12}$$

We conjecture that it is also self-concordant based on the similarity and the fact that our extensive numerical tests were successful.

In order to implement the cone (7) in the solver Hypatia, we need to provide the gradient, Hessian, and third-order derivative of the barrier function (12), and choose an initial point [20]. We remark that third-order derivatives are usually not required by primal-dual methods, and in particular not by the original Skajaa-Ye algorithm. Incorporating them is an innovation of Hypatia's algorithm.

3.1 Gradient, Hessian, and third-order derivative

To obtain the gradient, Hessian, and third-order derivative of the barrier function (12) we adapt the results of [26] to complex matrices, and combine them with the results of [27] for the relative entropy cone. In the code they are needed in vectorized form, as explained in Appendix A. Here we present them in the usual notation for clarity.

Let $g(\rho) = -H(\mathcal{L}(\rho))$ for some positive map $\mathcal{L}$. Then its gradient is

$$\nabla_\rho g(\rho) = \mathcal{L}^\dagger(\mathbb{1} + \log(\mathcal{L}(\rho))), \tag{13}$$

where $\mathcal{L}^\dagger$ is the adjoint of $\mathcal{L}$. Its Hessian is the linear operator

$$\nabla^2_{\rho,\rho} g(\rho) = \xi \mapsto \mathcal{L}^\dagger\Big(U(\Gamma^{[1]}(\Lambda) \odot (U^\dagger \mathcal{L}(\xi) U))U^\dagger\Big), \tag{14}$$

where $\odot$ is the elementwise (Schur) product, $\mathcal{L}(\rho) = U\Lambda U^\dagger$ is a diagonalization of $\mathcal{L}(\rho)$, and $\Gamma^{[1]}(\Lambda)$ is defined as

$$\Gamma^{[1]}(\Lambda)_{ij} = \begin{cases} \frac{\log(\lambda_i) - \log(\lambda_j)}{\lambda_i - \lambda_j}, & \lambda_i \neq \lambda_j \\ \lambda_i^{-1}, & \lambda_i = \lambda_j \,, \end{cases} \tag{15}$$

where $\lambda_i = \Lambda_{ii}$. Its third-order derivative applied to ξ, ξ is

$$\nabla^3_{\rho,\rho,\rho} g(\rho)[\xi, \xi] = \mathcal{L}^\dagger(U_\mathcal{L} M_\mathcal{L}(\xi) U_\mathcal{L}^\dagger)\,, \tag{16}$$

where

$$M_\mathcal{L}(\xi)_{\,i,j} = 2 \sum_k \tilde{\xi}_{i,k} \tilde{\xi}_{k,j} \Gamma^{[2]}_{i,j,k}(\Lambda_\mathcal{L})\,, \tag{17}$$

with $\tilde{\xi} = U_\mathcal{L}^\dagger \mathcal{L}(\xi) U_\mathcal{L}$ and

$$\Gamma^{[2]}_{i,j,k}(\Lambda) = \begin{cases} \frac{\Gamma^{[1]}_{i,j}(\Lambda) - \Gamma^{[1]}_{i,k}(\Lambda)}{\lambda_j - \lambda_k}, & \lambda_j \neq \lambda_k \\ \frac{\Gamma^{[1]}_{i,j}(\Lambda) - \Gamma^{[1]}_{j,j}(\Lambda)}{\lambda_i - \lambda_j}, & \lambda_i \neq \lambda_j = \lambda_k \\ -\lambda_i^{-2}/2, & \lambda_i = \lambda_j = \lambda_k\,. \end{cases} \tag{18}$$

The gradient and Hessian of the term $-\operatorname{logdet}(\rho)$ are well-known to be $-\rho^{-1}$ and $\xi \mapsto \rho^{-1}\xi\rho^{-1}$, respectively [28]. The third order derivative can be easily calculated to be

$$(\xi, \zeta) \mapsto -\rho^{-1}\xi\rho^{-1}\zeta\rho^{-1} - \rho^{-1}\zeta\rho^{-1}\xi\rho^{-1}. \tag{19}$$

Putting everything together, the gradient of the barrier $f(h, \rho)$ is given by

$$\nabla_h f(h,\rho) = -\frac{1}{u}, \tag{20}$$

$$\nabla_\rho f(h,\rho) = -\frac{1}{u}\nabla_\rho u - \rho^{-1}, \tag{21}$$

where $u = h + H(\widehat{\mathcal{G}}(\rho)) - H(\widehat{\mathcal{Z}}(\rho))$ and $\nabla_\rho u = -\widehat{\mathcal{G}}^\dagger(\mathbb{1} + \log(\widehat{\mathcal{G}}(\rho))) + \widehat{\mathcal{Z}}^\dagger(\mathbb{1} + \log(\widehat{\mathcal{Z}}(\rho)))$, and the Hessian by

$$\nabla^2_{h,h} f(h,\rho) = \frac{1}{u^2}, \tag{22}$$

$$\nabla^2_{h,\rho} f(h,\rho) = \frac{1}{u^2}\nabla_\rho u, \tag{23}$$

$$\nabla^2_{\rho,\rho} f(h,\rho) = \xi \mapsto \frac{1}{u^2}\operatorname{tr}\left[(\nabla_\rho u)\xi\right]\nabla_\rho u - \frac{1}{u}\nabla^2_{\rho,\rho} u[\xi] + \rho^{-1}\xi\rho^{-1}, \tag{24}$$

where

$$\nabla^2_{\rho,\rho} u[\xi] = -\widehat{\mathcal{G}}^\dagger\left(U_\mathcal{G}(\Gamma^{[1]}(\Lambda_\mathcal{G}) \odot (U_\mathcal{G}^\dagger \widehat{\mathcal{G}}(\xi) U_\mathcal{G}))U_\mathcal{G}^\dagger\right) + \widehat{\mathcal{Z}}^\dagger\left(U_\mathcal{Z}(\Gamma^{[1]}(\Lambda_\mathcal{Z}) \odot (U_\mathcal{Z}^\dagger \widehat{\mathcal{Z}}(\xi) U_\mathcal{Z}))U_\mathcal{Z}^\dagger\right), \tag{25}$$

and $\widehat{\mathcal{G}}(\rho) = U_\mathcal{G}\Lambda_\mathcal{G}U_\mathcal{G}^\dagger$ and $\widehat{\mathcal{Z}}(\rho) = U_\mathcal{Z}\Lambda_\mathcal{Z}U_\mathcal{Z}^\dagger$ are diagonalizations of $\widehat{\mathcal{G}}(\rho)$ and $\widehat{\mathcal{Z}}(\rho)$.

The third order derivatives are given by:

$$\nabla^3_{h,h,h} f(h,\rho) = -\frac{2}{u^3}, \tag{26}$$

$$\nabla^3_{h,h,\rho} f(h,\rho) = -\frac{2}{u^3}\nabla_\rho u, \tag{27}$$

$$\nabla^3_{h,\rho,\rho} f(h,\rho) = \xi \mapsto -\frac{2}{u^3}\operatorname{tr}\left[(\nabla_\rho u)\xi\right]\nabla_\rho u + \frac{1}{u^2}\nabla^2_{\rho,\rho} u[\xi], \tag{28}$$

$$\begin{aligned}\nabla^3_{\rho,\rho,\rho} f(h,\rho) = (\xi,\zeta) \mapsto &-\frac{2}{u^3}\operatorname{tr}\left[(\nabla_\rho u)\xi\right]\operatorname{tr}\left[(\nabla_\rho u)\zeta\right]\nabla_\rho u + \frac{1}{u^2}\operatorname{tr}\left[(\nabla^2_{\rho,\rho} u[\zeta])\xi\right]\nabla_\rho u \\ &+ \frac{1}{u^2}\operatorname{tr}\left[(\nabla_\rho u)\xi\right]\nabla^2_{\rho,\rho} u[\zeta] + \frac{1}{u^2}\operatorname{tr}\left[(\nabla_\rho u)\zeta\right]\nabla^2_{\rho,\rho} u[\xi] \\ &- \frac{1}{u}\nabla^3_{\rho,\rho,\rho} u[\xi,\zeta] - (\rho^{-1}\xi\rho^{-1}\zeta\rho^{-1} + \rho^{-1}\zeta\rho^{-1}\xi\rho^{-1}).\end{aligned} \tag{29}$$

In particular, applying $\nabla^3_{\rho,\rho,\rho} f(h,\rho)$ to the point $[\xi, \xi]$ gives

$$\begin{aligned}\nabla^3_{\rho,\rho,\rho} f(h,\rho)[\xi,\xi] = &-\frac{2}{u^3}\left(\operatorname{tr}\left[(\nabla_\rho u)\xi\right]\right)^2\nabla_\rho u + \frac{1}{u^2}\operatorname{tr}\left[(\nabla^2_{\rho,\rho} u[\xi])\xi\right]\nabla_\rho u \\ &+ \frac{2}{u^2}\operatorname{tr}\left[(\nabla_\rho u)\xi\right]\nabla^2_{\rho,\rho} u[\xi] - \frac{1}{u}\nabla^3_{\rho,\rho,\rho} u[\xi,\xi] \\ &- 2\rho^{-1}\xi\rho^{-1}\xi\rho^{-1},\end{aligned} \tag{30}$$

where

$$\nabla^3_{\rho,\rho,\rho} u[\xi,\xi] = -\widehat{\mathcal{G}}^\dagger(U_\mathcal{G} M_\mathcal{G}(\xi) U_\mathcal{G}^\dagger) + \widehat{\mathcal{Z}}^\dagger(U_\mathcal{Z} M_\mathcal{Z}(\xi) U_\mathcal{Z}^\dagger), \tag{31}$$

and $M_\mathcal{G}(\xi)$ and $M_\mathcal{Z}(\xi)$ are the analogues of (17).

3.2 Initial point

The initial point can in principle be chosen to be any point in the interior of the cone [15], but following [20, 29] we'd like to use the central point, as it provides better performance. It is defined as the minimizer of

$$\min_{h,\rho} f(h,\rho) + \frac{1}{2}\|(h,\rho)\|_2^2, \tag{32}$$

where $f(h,\rho)$ is our barrier function (12), and $\|(h,\rho)\|_2^2 = h^2 + \langle\rho,\rho\rangle$. The minimizer is obtained by taking the gradient of the objective and setting it to zero:

$$(h,\rho) = -\nabla f(h,\rho). \tag{33}$$

This results in the pair of equations

$$h = \frac{1}{u}, \tag{34}$$

$$\rho = \frac{1}{u}\nabla_\rho\, u + \rho^{-1}, \tag{35}$$

where we are using the gradient from equations (20)–(21).

Let now $u = h - D$, where $D = -H(\widehat{\mathcal{G}}(\rho)) + H(\widehat{\mathcal{Z}}(\rho))$. We get then[2]

$$h = \frac{D}{2} + \sqrt{1 + \frac{D^2}{4}}, \tag{36}$$

$$\rho - \rho^{-1} = h\Big(-\widehat{\mathcal{G}}^\dagger(\mathbb{1} + \log(\widehat{\mathcal{G}}(\rho))) + \widehat{\mathcal{Z}}^\dagger(\mathbb{1} + \log(\widehat{\mathcal{Z}}(\rho)))\Big). \tag{37}$$

While we have been unable to obtain a general solution of these equations, we note that often the maps satisfy $\widehat{\mathcal{G}}^\dagger(\mathbb{1} + \log(\widehat{\mathcal{G}}(\mathbb{1}))) = \widehat{\mathcal{Z}}^\dagger(\mathbb{1} + \log(\widehat{\mathcal{Z}}(\mathbb{1})))$. In this case $\rho = \mathbb{1}$ will satisfy equation (37) for any value of h, which we can then choose to be given by equation (36). We use therefore these values as our initial point.

Note that even when equation (37) is not satisfied this point is still in the interior of the cone, as $\mathbb{1}$ is full rank and $\frac{D}{2} + \sqrt{1 + \frac{D^2}{4}} > D$, and thus is a valid starting point.

4 Examples

4.1 BB84

In order to illustrate the technique let us start with a toy example, the computation of the key rate for the entanglement based version of the BB84 protocol, using as estimated parameters the qubit error rates (QBERs) in the X and Z bases q_x and q_z. If we use only the Z basis to generate key, the key map $\mathcal{G}$ can be taken to be identity, and the decoherence map $\mathcal{Z}$ is $\rho \mapsto \sum_{i=0}^{1}(|i\rangle\langle i| \otimes \mathbb{1})\rho(|i\rangle\langle i| \otimes \mathbb{1})$. The analytical expression for $H(A|E)$ is $1 - h(q_x)$, where h is the binary entropy.

In the case where $(q_x, q_z) \in (0,1)^{\times 2}$ there is a full rank state compatible with the constraints and the support of the range of $\mathcal{G}$ and $\mathcal{Z}$ is the full space, so no facial reduction is needed. Then $\widehat{\mathcal{G}} = \mathcal{G}$ and $\widehat{\mathcal{Z}} = \mathcal{Z} \circ \mathcal{G}$, and the conic program is

$$\begin{aligned} &\min_{h,\rho}\ h \\ \text{s.t.}\quad &\operatorname{tr}(\rho) = 1, \quad \operatorname{tr}(Q_x\rho) = q_x, \quad \operatorname{tr}(Q_z\rho) = q_z, \\ &(h,\rho) \in \mathcal{K}_{\text{QKD}}^{\widehat{\mathcal{G}},\widehat{\mathcal{Z}}}, \end{aligned} \tag{38}$$

where Q_x and Q_z are the projectors that produce the QBERs.

In the case where $q_z = 0$ and $q_x \in (0,1)$ the support of the feasible ρ is $\operatorname{span}\{|\phi^+\rangle, |\phi^-\rangle\}$. Letting $V = |\phi^+\rangle\langle 0| + |\phi^-\rangle\langle 1|$ we can write $\rho = V\sigma V^\dagger$ for a 2×2 matrix σ. The constraint $\operatorname{tr}(V^\dagger Q_z V\sigma) = 0$ becomes tautological and is dropped, and the constraint $\operatorname{tr}(V^\dagger Q_x V\sigma) = q_x$ is reduced to $\operatorname{tr}(|1\rangle\langle 1|\sigma) = q_x$. The map $\sigma \mapsto V\sigma V^\dagger$ has range supported on the range of V, namely $\operatorname{span}\{|\phi^+\rangle, |\phi^-\rangle\}$, so to restrict it there we simply remove the isometry, obtaining $\widehat{\mathcal{G}}(\sigma) = \sigma$. The map $\sigma \mapsto \mathcal{Z}(V\sigma V^\dagger)$ has range supported on $\operatorname{span}\{|00\rangle, |11\rangle\}$, so we restrict it there, choosing the Kraus operators of $\widehat{\mathcal{Z}}$ to be $\{|0\rangle\langle +|, |1\rangle\langle -|\}$.

In the case where $q_x = 0$ and $q_z \in (0,1)$ the support of the feasible ρ is $\operatorname{span}\{|\phi^+\rangle, |\psi^+\rangle\}$. Letting $V = |\phi^+\rangle\langle 0| + |\psi^+\rangle\langle 1|$ we can write $\rho = V\sigma V^\dagger$ for a 2×2 matrix σ. The constraint

[2]The negative solution for h is excluded as it is outside the cone.

$\mathrm{tr}(V^\dagger Q_x V\sigma) = 0$ becomes tautological and is dropped, and the constraint $\mathrm{tr}(V^\dagger Q_z V\sigma) = q_z$ is reduced to $\mathrm{tr}(|1\rangle\langle 1|\sigma) = q_z$. To reduce the map $\sigma \mapsto V\sigma V^\dagger$ we again simply remove the isometry, obtaining $\widehat{\mathcal{G}}(\sigma) = \sigma$. The map $\sigma \mapsto \mathcal{Z}(V\sigma V^\dagger)$ this time has full rank range, so its reduction is simply itself, $\widehat{\mathcal{Z}}(\sigma) = \mathcal{Z}(V\sigma V^\dagger)$.

In the case where $q_x = q_z = 0$ the only feasible ρ is $|\phi^+\rangle\langle\phi^+|$ so there's nothing to optimize.

4.2 DMCV

A more interesting case of facial reduction is when a POVM $\{E_i\}_{i=0}^{n-1}$ is used to generate key. This is the case for example in discrete-modulated continuous variable (DMCV) QKD [30]. For simplicity, assume that the estimated probabilities are compatible with a full-rank state, so that facial reduction is only needed for the maps $\mathcal{G}$ and $\mathcal{Z}$.

Let $V = \sum_{i=0}^{n-1} \mathbb{1}_A \otimes \sqrt{E_i}_B \otimes |i\rangle_a$ be the standard Naimark dilation of the POVM[3], where a is an ancilla subsystem. Then $\mathcal{G}(\rho) = V\rho V^\dagger$, and $\mathcal{Z}$ has Kraus operators $\{\mathbb{1}_{AB} \otimes |i\rangle\langle i|_a\}_{i=0}^{n-1}$. As before, since $\mathcal{G}$ is just an isometry, the reduction consists simply of removing it, and $\widehat{\mathcal{G}}(\rho) = \rho$. The Kraus operators of $\mathcal{Z} \circ \mathcal{G}$ are $\{\mathbb{1}_A \otimes \sqrt{E_i}_B \otimes |i\rangle_a\}_{i=0}^{n-1}$. It is easy to see that if all E_i are full rank then $\mathcal{Z} \circ \mathcal{G}$ has full rank range, and $\widehat{\mathcal{Z}} = \mathcal{Z} \circ \mathcal{G}$. Otherwise further reduction is needed (or starting with a more appropriate Naimark dilation).

For the purpose of benchmarking we implemented here the heterodyne protocol from [30] (a more sophisticated application of our technique to DMCV is presented in [31]). In it the POVMs are in fact full rank and the facial reduction is as described in the previous paragraph. The main parameter of the protocol is the number of photons at which the Fock space is cut off, N_c. The quantum state ρ has dimension $4(N_c+1)$, and the Kraus operators of $\widehat{\mathcal{Z}}$ are $16(N_c+1) \times 4(N_c+1)$. As it is a prepare-and-measure protocol, it also has constraints on the partial trace of ρ, which in general may introduce null eigenvalues and necessitate further facial reduction. Here it is not the case.

4.3 MUB protocol

In the MUB protocol introduced in [7, 8], Alice and Bob measure a complete set of MUBs for some prime d, with Bob's bases being the transpose of Alice's, and estimate the probability of getting equal results. As in [10], here there is no such limitations of using prime dimensions, equal results, or even exact MUBs.

As it is an entanglement-based protocol and we are generating key in a single basis, the key map $\mathcal{G}$ can be taken to be identity, and the decoherence map $\mathcal{Z}$ to have Kraus operators $\{|i\rangle\langle i| \otimes \mathbb{1}\}_{i=0}^{d-1}$. When the estimated statistics are compatible with a full-rank state the facial reduction is trivial: $\widehat{\mathcal{G}} = \mathcal{G}$ and $\widehat{\mathcal{Z}} = \mathcal{Z} \circ \mathcal{G}$.

4.4 Overlapping bases protocol

In the overlapping bases protocol introduced in [10], Alice and Bob measure only nearest-neighbour superpositions, with again Bob's bases being the transpose of Alice's. $\mathcal{G}$ and $\mathcal{Z}$ are the same as in the MUB protocol, and the facial reduction is also trivial the estimated statistics are compatible with a full-rank state: $\widehat{\mathcal{G}} = \mathcal{G}$ and $\widehat{\mathcal{Z}} = \mathcal{Z} \circ \mathcal{G}$.

We note, however, that in [10] the protocol was restricted to even $d \geq 2$, whereas here we use it also for odd d.

4.5 Dealing with experimental data

As argued in [10], in order to deal with experimental data one cannot naïvely set the probabilities $\mathbf{p}$ to be equal to the measured relative frequencies $\mathbf{f}$. Instead, one should estimate the covariance matrix Σ, and minimize $H(A|E)$ over the desired confidence region, approximated as the intersection of the ellipsoid $\left\|\Sigma^{-\frac{1}{2}}(\mathbf{p}-\mathbf{f})\right\|_2 \leq \chi$ with the set of parameters corresponding to valid quantum states.

[3]Note that here we are applying the POVMs on Bob's side, as is standard in DMCV.

One can do that with a simple modification of our conic program (8):

$$\begin{aligned} &\min_{h,\sigma,\mathbf{p}} \quad h \\ \text{s.t.} \quad &\operatorname{tr}(\sigma) = 1, \quad \operatorname{tr}(\mathbf{F}\sigma) = \mathbf{p}, \quad \left\|\Sigma^{-\frac{1}{2}}(\mathbf{p}-\mathbf{f})\right\|_2 \leq \chi, \\ &(h,\sigma) \in \mathcal{K}_{\mathrm{QKD}}^{\widehat{\mathcal{G}},\widehat{\mathcal{Z}}} \end{aligned} \tag{39}$$

where the constraint we added is a second-order-cone constraint, which is supported by almost every conic solver in existence, and in particular the one we are using, Hypatia.

5 Numerical results

5.1 Performance

In order to illustrate the performance of our technique, we ran the conic program (8) and compared the running time to the Gauss-Newton technique [9] and the Gauss-Radau technique [10]. We didn't compare to the Frank-Wolfe technique from [11] because [9] already demonstrated superior performance. We didn't compare to the technique from [32] because they only use the vanilla relative entropy cone; as such they cannot perform facial reduction and are unable to handle the QKD problem in full generality.

The protocols we benchmarked are the DMCV, MUB and overlap protocols described in subsections 4.2, 4.3 and 4.4. We chose them because they are currently of high experimental and theoretical interest [31, 33–35], and because they have a free parameter controlling the dimension that allows us to see how the solution time scales.

For the DMCV protocol we used the same parameters as in [9]: amplitude of the coherent states $\alpha = 0.35$, noise $\xi = 0.05$, distance $L = 60$, and transmittance $\eta = 10^{-0.02L}$. For the MUB and overlap protocols we used the probabilities from the isotropic state $\rho_{\mathrm{iso}}(v) = v|\phi^+\rangle\langle\phi^+| + (1-v)\mathbb{1}/d^2$ with visibility $v = 0.95$. For the MUB protocol we computed the probabilities using all bases, included the complex ones, whereas for the overlap protocol we restricted ourselves to the real ones. We did this in order to illustrate an additional advantage of our technique: since it uses standard conic optimization, we can avail ourselves of standard symmetrization techniques to show that we can optimize over real variables only, as done in [10]. This provides an additional boost in performance.

The calculations were done on an AMD Ryzen Threadripper Pro 5955WX with 4 GHz and 16 cores, on a machine with 512 GiB RAM running Ubuntu Linux 22.04. Our code was run with Julia 1.11.1 and the modeller JuMP [21], and the two other techniques with MATLAB 2023b. For the Gauss-Radau technique we used the solver MOSEK 10.1 [36] and the modeller YALMIP [37]. In all cases we have only reported the time taken by the solver, discounting the time taken to set up the problem, and in the Gauss-Newton case the time taken to do facial reduction numerically.

The results are shown in Figures 1, 2, and 3.

5.2 Precision

In order to obtain results with higher precision, one might try setting tighter tolerances for the conic solver. This is however not fruitful, as very quickly one meets fundamental obstacles due to the limited precision with which the standard 64-bit floating point numbers can perform computations or even represent the problem parameters. For this reason we have used instead different implementations of floating point numbers that provide more bits of precision. This is easy due to Julia's flexible type system, which is fully taken advantage of by the solver Hypatia and the modeller JuMP; in order to use a different type we simply need to give it as a parameter. Hypatia automatically sets the appropriate tolerances for each type. We have used the following four types:

`Float64` Default 64-bit floating point number implementing the IEEE 754 standard, known as double. Has 53 bits of precision.

`Float128` 128-bit floating point number implementing the IEEE 754 standard, known as quadruple. Has 113 bits of precision. Provided by the package Quadmath [38], which wraps the GCC library libquadmath.

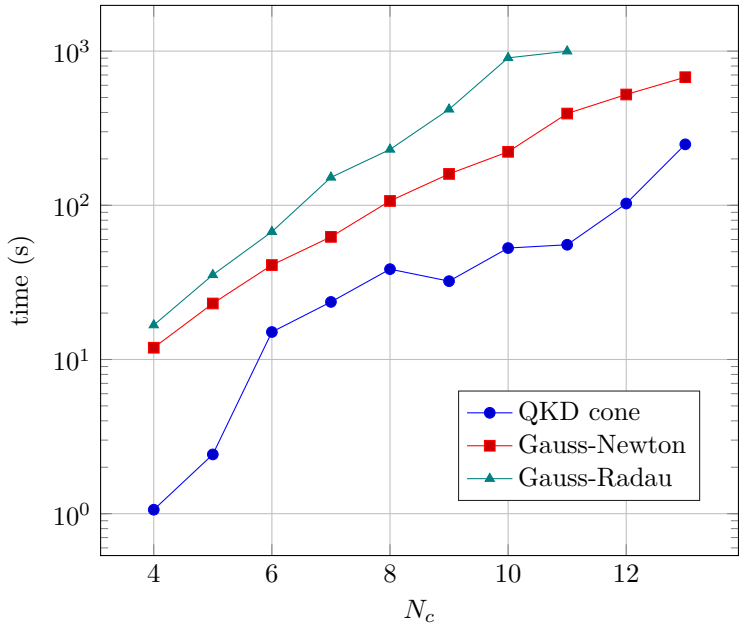

Figure 1: Running time in seconds (logarithmic scale) as a function of the photon cut-off number for the DMCV protocol.

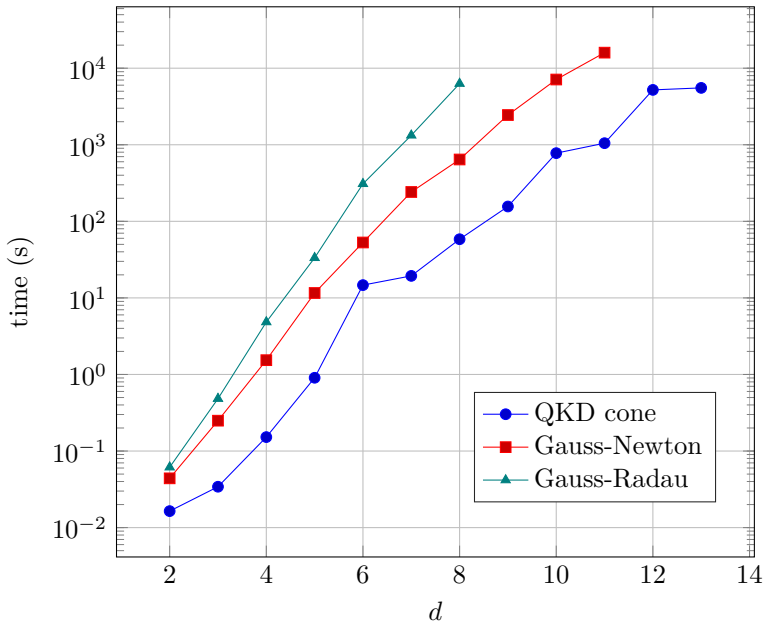

Figure 2: Running time in seconds (logarithmic scale) as a function of the local state dimension for the MUB protocol. Note that for $d = 6, 10$, and 12 the bases used are only roughly unbiased.

`Double64` Non-standard implementation of a 128-bit floating point number as a pair of 64-bit floating point numbers, known as double-double. Much faster than `Float128`, but has smaller precision and smaller range of exponent. Has 106 bits of precision. Provided by the package DoubleFloats [39].

`BigFloat` Arbitrary precision floating-point number. We used the default settings, with which it occupies 640 bits of memory and provides 256 bits of precision. Provided by the Julia standard library, wrapping the GNU MPFR library [40].

In order to evaluate the error for which type, we solved the conic program (8) for the protocols we

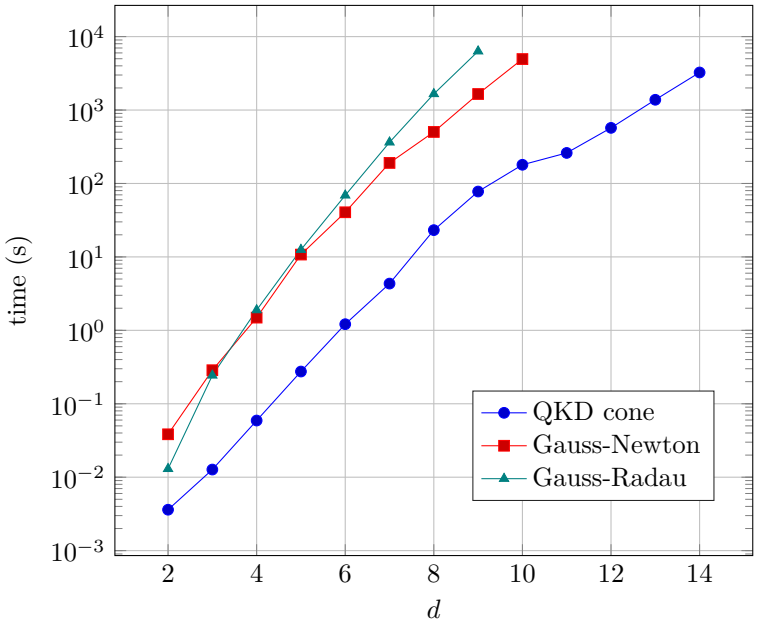

Figure 3: Running time in seconds (logarithmic scale) as a function of the local state dimension for the overlap protocol.

know the analytical answers: BB84 4.1 and MUB 4.3. The results are shown in Table 1.

Although the analytical answer is also known for the DMCV protocol 4.2 for the case $\xi = 0$, a comparison is not meaningful: after facial reduction there is a unique state compatible with the constraints, so there's nothing to optimize. We have nevertheless provided the analytical answer and the facial reduction in the accompanying source code for the interested reader.

	BB84	MUB
`Float64`	5.4×10^{-8}	5.7×10^{-8}
`Double64`	1.4×10^{-13}	1.6×10^{-12}
`Float128`	1.7×10^{-14}	7.7×10^{-14}
`BigFloat`	7.9×10^{-32}	1.6×10^{-31}

Table 1: Absolute difference between analytical $H(A|E)$ and the one computed via the conic program (8) for various floating point implementations, for the BB84 and MUB protocols. For the BB84 protocol we used the parameters $q_x = q_z = 0.025$, and for the MUB protocol we used dimension 3 with 4 bases and the probabilities of an isotropic state with visibility $v = 0.95$.

Note that in general a number that has a finite decimal expansion does not have a finite binary expansion. For example, $0.95_{10} = 0.1111\overline{0011}_2$. Therefore, even when giving input parameters that at first sight seem exact, one often incurs in truncation errors. Therefore, in order to take advantage of higher precision types, the input parameters also need to use them[4]. For this reason our example codes don't require the type desired for the computation to be specified, but rather read it from the type of the input parameters.

It's important to emphasize that the increased precision comes at the cost of a much longer running time. Not only the elementary operations take longer, but also the number of iterations increases in order to meet the tighter tolerances.

[4]One must also be careful to write 0.95 as (e.g.) `Double64(95)/100` instead of `Double64(0.95)`, as the latter first computes 0.95 as a `Float64` before converting it to `Double64`.

6 Conclusion

In this work we have introduced a new technique for obtaining the asymptotic key rate in quantum key distribution. It is based on defining a new convex cone for the key rate problem and optimizing over it with standard conic optimization methods. It provides a dramatic improvement in performance with respect to previous techniques. Moreover, it is flexible and easy to use, as one can freely combine it with other convex cones, and it is interfaced through a convenient modelling language, JuMP.

As a primal-dual method, it directly provides a lower bound through the value of the dual objective. However, we lack an explicit expression for the dual cone, which stops us from converting the dual minimizer into an independent witness for the lower bound. One can nevertheless construct such a witness from the primal minimizer as described in [11] and done for example in [31].

The most pressing issue is to adapt the method to finding finite key rates. Although one can use the asymptotic key rate to obtain a finite key rate via the (generalized) entropy accumulation theorem [41, 42], as done for example in [31], such bounds are known to be loose. Tight bounds are provided by more recent techniques [43, 44], that however require optimization over Rényi entropies. It would be interesting to apply conic methods to this case.

7 Code availability

The implementation of the cone introduced in this paper and the code for the examples shown are available in `https://github.com/araujoms/ConicQKD.jl`.

8 Related work

While conducting this research, we found that He et al. [45] had independently applied the Skajaa and Ye algorithm to the problem of QKD using a specialized cone, and proved the self-concordance of the barrier function (12). In version 0.2 of our code we implemented the technique for inverting the Hessian described in their Appendix B.2.

9 Acknowledgements

The research of A.G.L., P.V.P., M.C.C. and M.A. was supported by the European Union–Next Generation UE/MICIU/Plan de Recuperación, Transformación y Resiliencia/Junta de Castilla y León. M.A was also supported by the Spanish Agencia Estatal de Investigación, Grant No. RYC2023-044074-I. We thank Chris Coey and Lea Kapelevich for help with Hypatia's code and useful discussions.

References

[1] N. Gisin, G. Ribordy, W. Tittel and H. Zbinden, 'Quantum cryptography', Reviews of Modern Physics **74**, 145–195 (2002), arXiv:quant-ph/0101098.

[2] V. Scarani, H. Bechmann-Pasquinucci, N. J. Cerf, M. Dušek, N. Lütkenhaus and M. Peev, 'The security of practical quantum key distribution', Reviews of Modern Physics **81**, 1301–1350 (2009), arXiv:0802.4155 [quant-ph].

[3] F. Xu, X. Ma, Q. Zhang, H.-K. Lo and J.-W. Pan, 'Secure quantum key distribution with realistic devices', Reviews of Modern Physics **92**, 025002 (2020), arXiv:1903.09051 [quant-ph].

[4] S. Pirandola et al., 'Advances in quantum cryptography', Advances in Optics and Photonics **12**, 1012 (2020), arXiv:1906.01645 [quant-ph].

[5] C. H. Bennett and G. Brassard, 'Quantum cryptography: public key distribution and coin tossing', Theoretical Computer Science **560**, (reprint), 7–11 (1984).

[6] D. Bruß, 'Optimal Eavesdropping in Quantum Cryptography with Six States', Physical Review Letters **81**, 3018–3021 (1998), arXiv:quant-ph/9805019.

[7] N. J. Cerf, M. Bourennane, A. Karlsson and N. Gisin, 'Security of quantum key distribution using d-level systems', Physical Review Letters **88**, 127902 (2002), arXiv:quant-ph/0107130.

[8] L. Sheridan and V. Scarani, 'Security proof for quantum key distribution using qudit systems', Physical Review A **82**, 030301(R) (2010), arXiv:1003.5464 [quant-ph].

[9] H. Hu, J. Im, J. Lin, N. Lütkenhaus and H. Wolkowicz, 'Robust interior point method for quantum key distribution rate computation', Quantum **6**, 792 (2022), arXiv:2104.03847 [quant-ph].

[10] M. Araújo, M. Huber, M. Navascués, M. Pivoluska and A. Tavakoli, 'Quantum key distribution rates from semidefinite programming', Quantum **7**, 1019 (2023), arXiv:2211.05725 [quant-ph].

[11] A. Winick, N. Lütkenhaus and P. J. Coles, 'Reliable numerical key rates for quantum key distribution', Quantum **2**, 77 (2018), arXiv:1710.05511 [quant-ph].

[12] Y. Nesterov, M. Todd and Y. Ye, 'Infeasible-start primal-dual methods and infeasibility detectors for nonlinear programming problems', Mathematical Programming **84**, 227–267 (1999).

[13] L. Tunçel, 'Generalization of primal-dual interior-point methods to convex optimization problems in conic form', Foundations of Computational Mathematics **1**, 229–254 (2001).

[14] Y. Nesterov, 'Towards non-symmetric conic optimization', Optimization Methods and Software **27**, 893–917 (2012).

[15] A. Skajaa and Y. Ye, 'A homogeneous interior-point algorithm for nonsymmetric convex conic optimization', Mathematical Programming **150**, 391–422 (2015).

[16] D. Papp and S. Yıldız, *On "A homogeneous interior-point algorithm for non-symmetric convex conic optimization"*, 2017, arXiv:1712.00492 [math.OC].

[17] D. Drusvyatskiy and H. Wolkowicz, 'The many faces of degeneracy in conic optimization', Foundations and Trends in Optimization **3**, 77–170 (2017), arXiv:1706.03705 [quant-ph].

[18] J. Bezanson, A. Edelman, S. Karpinski and V. B. Shah, 'Julia: a fresh approach to numerical computing', SIAM Review **59**, 65–98 (2017), arXiv:1411.1607 [cs.MS].

[19] C. Coey, L. Kapelevich and J. P. Vielma, 'Solving natural conic formulations with Hypatia.jl', INFORMS Journal on Computing **34**, 2686–2699 (2022), arXiv:2005.01136 [math.OC].

[20] C. Coey, L. Kapelevich and J. P. Vielma, 'Performance enhancements for a generic conic interior point algorithm', Mathematical Programming Computation **15**, 53–101 (2023), arXiv:2107.04262 [math.OC].

[21] M. Lubin, O. Dowson, J. Dias Garcia, J. Huchette, B. Legat and J. P. Vielma, 'JuMP 1.0: Recent improvements to a modeling language for mathematical optimization', Mathematical Programming Computation **15**, 581–589 (2023), arXiv:2206.03866 [cs.PL].

[22] I. Devetak and A. Winter, 'Distillation of secret key and entanglement from quantum states', Proceedings of the Royal Society of London Series A **461**, 207–235 (2005), arXiv:quant-ph/0306078.

[23] P. J. Coles, 'Unification of different views of decoherence and discord', Physical Review A **85**, 042103 (2012), arXiv:1110.1664 [quant-ph].

[24] Y. Nesterov, *Lectures on convex optimization* (Springer Cham, 2018).

[25] H. Fawzi and J. Saunderson, 'Optimal self-concordant barriers for quantum relative entropies', SIAM Journal on Optimization **33**, 2858–2884 (2023), arXiv:2205.04581 [math.OC].

[26] L. Faybusovich and C. Zhou, 'Long-step path-following algorithm for quantum information theory: some numerical aspects and applications', Numerical Algebra, Control and Optimization **12**, 445–467 (2022), arXiv:1906.00037 [math.OC].

[27] C. Coey, L. Kapelevich and J. P. Vielma, *Hypatia cones reference*, (2024) https://github.com/jump-dev/Hypatia.jl/wiki/files/coneref.pdf.

[28] Y. E. Nesterov and M. J. Todd, 'Self-scaled barriers and interior-point methods for convex programming', Mathematics of Operations Research **22**, 1–42 (1997).

[29] J. Dahl and E. D. Andersen, 'A primal-dual interior-point algorithm for nonsymmetric exponential-cone optimization', Mathematical Programming **194**, 341–370 (2022).

[30] J. Lin, T. Upadhyaya and N. Lütkenhaus, 'Asymptotic security analysis of discrete-modulated continuous-variable quantum key distribution', Physical Review X **9**, 041064 (2019), arXiv:1905.10896 [quant-ph].

[31] C. Pascual-García, S. Bäuml, M. Araújo, R. Liss and A. Acín, 'Improved finite-size key rates for discrete-modulated continuous variable quantum key distribution under coherent attacks', Physical Review A **111**, 022610 (2025), arXiv:2407.03087 [quant-ph].

[32] M. Karimi and L. Tunçel, 'Efficient implementation of interior-point methods for quantum relative entropy', INFORMS Journal on Computing **37**, 3–21 (2024), arXiv:2312.07438 [quant-ph].

[33] L. Bulla et al., 'Distribution of genuine high-dimensional entanglement over 10.2 km of noisy metropolitan atmosphere', Phys. Rev. A **107**, L050402 (2023), arXiv:2301.05724 [quant-ph].

[34] O. Lib, K. Sulimany, M. Araújo, M. Ben-Or and Y. Bromberg, *High-dimensional quantum key distribution using a multi-plane light converter*, 2024, arXiv:2403.04210 [quant-ph].

[35] I. W. Primaatmaja, W. Y. Kon and C. Lim, *Discrete-modulated continuous-variable quantum key distribution secure against general attacks*, 2024, arXiv:2409.02630 [quant-ph].

[36] MOSEK ApS, *The MOSEK Optimization Suite 10.1.11*, https://docs.mosek.com/latest/intro/index.html (2023).

[37] J. Löfberg, 'YALMIP: a toolbox for modeling and optimization in MATLAB', in Proceedings of the CACSD conference, https://yalmip.github.io/ (2004), pp. 284–289.

[38] H. Hofstätter, S. Byrne and R. Smith, *Quadmath*, version 0.5.10, 2024, https://github.com/JuliaMath/Quadmath.jl.

[39] J. Sarnoff and JuliaMath, *DoubleFloats*, version 1.4.0, 2024, https://github.com/JuliaMath/DoubleFloats.jl.

[40] L. Fousse, G. Hanrot, V. Lefèvre, P. Pélissier and P. Zimmermann, 'MPFR: A multiple-precision binary floating-point library with correct rounding', ACM Trans. Math. Softw. **33**, 13–es (2007).

[41] F. Dupuis, O. Fawzi and R. Renner, 'Entropy accumulation', Communications in Mathematical Physics **379**, 867–913 (2020), arXiv:1607.01796 [quant-ph].

[42] T. Metger, O. Fawzi, D. Sutter and R. Renner, 'Generalised entropy accumulation', in 2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS) (Oct. 2022), arXiv:2203.04989 [quant-ph].

[43] T Van Himbeeck and P Brown, *Tight and general finite-size security of quantum key distribution*, (in preparation), 2024.

[44] A. Arqand, T. A. Hahn and E. Y.-Z. Tan, *Generalized Rényi entropy accumulation theorem and generalized quantum probability estimation*, 2024, arXiv:2405.05912 [quant-ph].

[45] K. He, J. Saunderson and H. Fawzi, *Exploiting structure in quantum relative entropy programs*, 2024, arXiv:2407.00241 [quant-ph].

A Vectorization

Let $\mathrm{vec} : \mathbb{C}^{d_O \times d_I} \to \mathbb{C}^{d_O d_I}$ be the col-major vectorization of a real or complex rectangular matrix. It is useful to represent it as

$$\mathrm{vec}(X) = \mathbb{1}_{d_I} \otimes X \,\mathrm{vec}(\mathbb{1}_{d_I}), \tag{40}$$

where $\mathrm{vec}(\mathbb{1}_{d_I}) = \sum_{i=0}^{d_I-1} |ii\rangle$. It is common to write $\mathrm{vec}(X)$ as $|X\rangle\!\rangle$. A useful identity is

$$\mathrm{vec}(ABC) = C^T \otimes A \,\mathrm{vec}(B). \tag{41}$$

For implementation purposes we work with a non-redundant vectorization for Hermitian matrices $\mathrm{svec}(X)$. The real and complex cases differ. In the real case, $\mathrm{svec} : \mathbb{R}^{d\times d} \to \mathbb{R}^{d(d+1)/2}$ is the col-major vectorization of the upper triangle, with the off-diagonals multiplied by $\sqrt{2}$. In the complex case, $\mathrm{svec} : \mathbb{C}^{d\times d} \to \mathbb{R}^{d^2}$ additionally splits the complex numbers into the real part and minus the imaginary part[5], storing them consecutively.

The factor of $\sqrt{2}$ multiplying the off-diagonals is necessary to ensure that

$$\langle X, Y\rangle = \langle \mathrm{vec}(X), \mathrm{vec}(Y)\rangle = \langle \mathrm{svec}(X), \mathrm{svec}(Y)\rangle \quad \forall X, Y. \tag{42}$$

Let V be the isometry[6] such that

$$\mathrm{vec}(X) = V \,\mathrm{svec}(X) \tag{43}$$

for all Hermitian X. Suppose you are interested in representing a linear function f as a matrix F such that

$$F \,\mathrm{vec}(X) = \mathrm{vec}(f(X)) \quad \forall X. \tag{44}$$

This is easy to do using identity (41) and linearity. If this function is additionally Hermitian-preserving we have that

$$V^\dagger F V \,\mathrm{svec}(X) = \mathrm{svec}(f(X)) \tag{45}$$

for all Hermitian X.

This is mainly useful for proving theorems, as a direct computation of $V^\dagger F V$ via this formula is inefficient. For the particular case of $f(X) = KXK^\dagger$ we implemented an efficient function to compute it, $\mathrm{skron}(K) = V^\dagger(\bar{K} \otimes K)V$.

[5] This is an arbitrary choice, made to coincide with the code.

[6] This is an abuse of notation since the operator is different in the real and complex cases. Also note that in the complex case V is additionally unitary.