SUPER-LINEAR CONVERGENCE IN THE P-ADIC QR-ALGORITHM

Avinash Kulkarni, Tristan Vaccon

SUPER-LINEAR CONVERGENCE IN THE P-ADIC QR-ALGORITHM

AVINASH KULKARNI AND TRISTAN VACCON

Abstract. The QR-algorithm is one of the most important algorithms in linear algebra. Its several variants make feasible the computation of the eigenvalues and eigenvectors of a numerical real or complex matrix, even when the dimensions of the matrix are enormous. The first adaptation of the QR-algorithm to local fields was given by the first author in 2019. However, in this version the rate of convergence is only linear and in some cases the decomposition into invariant subspaces is incomplete. We present a refinement of this algorithm with a super-linear convergence rate in many cases.

1. Introduction

Eigenvalues and eigenvectors are ubiquitous throughout mathematics and industrial applications. Much attention has been directed towards developing algorithms to compute the eigenvectors of a finite precision real or complex matrix, whether for the purposes of making new computations feasible in research or for a more efficient product in industry. Given the successes of eigenvector methods in numerical linear algebra, one can hope that exciting novel applications can come from the comparatively unexplored area of finite precision p-adic linear algebra. In analogy, we refer to the subject as *pnumerical linear algebra*.

The topic of pnumerical linear algebra was first addressed in the latter half of the 20th century [1,2]. Recently it has seen renewed interest [3–5]. A noteworthy application is a polynomial time algorithm for computing points on an algebraic curve over a finite field based on computing the characteristic polynomial of a p-adic matrix [6]. This is useful in practical cryptography to select curves with good properties for cryptosystems. Another application is in solving a 0-dimensional system of polynomial equations over $\mathbb{Q}_p$ [7,8].

The first method for computing the eigenvectors of a p-adic (or real) matrix M is the schoolbook algorithm, consisting of the following steps:

(1) Compute a Hessenberg form for M. (Optimization for step 4.)
(2) Compute the characteristic polynomial of M.
(3) Solve for the roots $\{\lambda_i\}$.
(4) Compute $\ker(M - \lambda_i I)$ for each i.
(4b) (For block Schur form:) Compute $\ker(M - \lambda_i I)^{d_i}$ for some d_i.

Over the reals, this is not the main algorithm used in practice since step (3) is numerically unstable. A similar difficulty is encountered p-adically, in that one needs to know the characteristic polynomial at the maximum possible p-adic precision in order to correctly compute the roots. In the worst case scenario, for an $n \times n$ input matrix given at N digits of precision in each entry, one needs to compute the characteristic polynomial using arithmetic with nN digits – for examples see Section 2.4. For practical considerations, one must be careful of the extra costs imposed by precision increases. Worse still, step 4b can fail to give the correct answer due to a lack of precision on the input (see Example 3.1.1). Unlike $\mathbb{R}$, p-adic fields admit algebraic extensions of arbitrarily large degree. Consequently, the cost of doing arithmetic in an extension is potentially much more severe.

To represent the finite precision of the input, we will say that $M \in \mathrm{M}_n(\mathbb{Q}_p)$ is known with (absolute) error $O(p^N)$ if we know the initial part $a_{-v}p^{-v} + \ldots + a_{N-1}p^{N-1} + O(p^N)$ of the p-adic expansion for each entry in the matrix M. We say that $A = B + O(p^N)$ if the p-adic expansion for every entry of $A - B$ up to the p^N term is 0. In Section 2, we discuss p-adic precision in more detail. We say that a

2010 *Mathematics Subject Classification.* 15A18 (primary), 11S05 (secondary).

Key words and phrases. QR-algorithm, symbolic-numeric, p-adic approximation, pnumerical linear algebra.

Avinash Kulkarni has been supported by the Simons Collaboration on Arithmetic Geometry, Number Theory, and Computation (Simons Foundation grant 550033) and by the Forschungsinitiative on Symbolic Tools at TU Kaiserslautern.

matrix is in *block Schur form* if it is block upper triangular and the characteristic polynomial of each diagonal block is irreducible. The main problem of this article is:

Problem 1.0.1. *Given an $n \times n$ matrix M over $\mathbb{Q}_p$, whose entries are known with error $O(p^N)$, compute a block Schur form T for M and a matrix U such that $MU = UT + O(p^N)$.*

Of course, by passing to the splitting field of the characteristic polynomial, we can convert a block Schur form to a Schur form by triangularizing each of the blocks. In this article, we choose to use only $\mathbb{Q}_p$-arithmetic. The benefit being that we procrastinate on doing expensive extension field arithmetic. When $M \in \mathrm{M}_n(\mathbb{Z}_p)$, it is possible to put M into a block Schur form via some $V \in \mathrm{GL}_n(\mathbb{Z}_p)$.

Theorem (3.0.2)**.** *Let $M \in \mathrm{M}_n(\mathbb{Z}_p)$ and let $\chi_M = f_1 \cdots f_r$ be a factorization in $\mathbb{Z}_p[t]$ where the factors are pairwise coprime in $\mathbb{Q}_p[t]$. Then there exists a $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ such that UMU^{-1} is block-triangular with r blocks; the j-th block accounts for the eigenvalues λ such that $f_j(\lambda) = 0$.*

Our method of proof is to combine standard arguments for the existence of canonical forms over a field with the notion of orthogonality, introduced in Schikhof [9] and discussed in Section 2. As an immediate corollary, we obtain a refinement of the decomposition of [5, Theorem 4.3.11].

Corollary (3.0.3, Newton decomposition)**.** *Let $M \in \mathrm{M}_n(\mathbb{Z}_p)$ and let $\nu_1 \leq \ldots \leq \nu_r$ be the distinct valuations of the eigenvalues of M. Then there exists a $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ such that UMU^{-1} is block-triangular with r diagonal blocks; the j-th block accounts exactly for the eigenvalues of valuation ν_j.*

We next show how to improve the iterative computation of the block Schur form introduced in [8]. Our main theorem is:

Theorem (5.1.1)**.** *Let $M \in \mathrm{M}_n(\mathbb{Z}_p)$ be a matrix whose entries are known with error $O(p^N)$. If the characteristic polynomial of M modulo p is square-free and factors completely then Algorithm 5.1 computes a Schur form T and a matrix $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ such that $MU = UT + O(p^N)$ in at most $\frac{2}{3}n^3 \log_2 N + o(n^3 \log_2 N)$ arithmetic operations in $\mathbb{Z}_p$ at N-digits of precision. In particular, T reveals all the eigenvalues of M with error $O(p^N)$. An additional $O(n^3)$ arithmetic operations in $\mathbb{Q}_p$ is then enough to compute a $\mathbb{Q}_p$-basis of eigenvectors with coefficients in $\mathbb{Z}_p$.*

If the assumptions of the theorem above are not met, our algorithm will attempt to use the accelerated convergence strategy anyway. The timings in Section 6 demonstrate a significant improvement over both the classical method and the basic QR-iteration in [8] in computing a weak block Schur form (see Definition 5.0.1), *even when the hypothesis on the characteristic polynomial is not satisfied.* Furthermore, the slowdown of convergence can be detected dynamically in Algorithm 4.3. Should this occur we fallback to running the QR-iteration with linear convergence, as in [8].

We describe the layout of the article. For the remainder of Section 1, we establish notation and then precisely state our results regarding Algorithm 4.3. In Section 2 we discuss the background needed in the article. In Section 3, we prove Theorem 3.0.2 and discuss the computation of sorted and size-sorted forms. In Section 4, we discuss the improved QR-iteration; here we give Algorithm 4.3. In Section 5, we combine our results to produce Algorithm 5.1 and we also prove Theorem 5.1.1. Finally, in Section 6 we discuss the implementation of our algorithm and give some timings.

1.1. **Notation.** We denote by '$*$' a wildcard ($\mathbb{Z}_p$-)integral entry or block of integral entries in a matrix. Generally, we will use the wildcard entries in upper-right blocks as they are not especially noteworthy in our analysis aside from the fact that they are integral. For a matrix M, we denote its left kernel by $\mathrm{lker}(M)$ and its characteristic polynomial by χ_M. If $\chi_M(t) \in \mathbb{Z}_p[t]$ we denote by $\chi_{M,p}$ the reduction of χ_M to the residue field. The standard basis vectors are denoted by $e_1, e_2, \ldots, e_n$. For a ring R, the ring of $n \times n$-matrices with entries in R is denoted $\mathrm{M}_n(R)$. The (i,j)-th entry of a matrix is denoted $A\{i,j\}$ and $A\{\bullet, j\}$ denotes the j-th column. Our choice of notation deviates from the standard to improve the readability of expressions like $\left|R_B^{(j)}\{1,1\}\right|$.

The p-adic absolute value is denoted by $|\cdot|$ and normalized so that $|p| = p^{-1}$, for a vector v we denote $\|v\| := \max_i |v_i|$, and for a matrix A we denote $\|A\| := \max_{i,j} |A\{i,j\}|$. For a polynomial $f := f_n x^n + \ldots + f_0$, we denote $\|f\| := \max_i |f_i|$.

For a matrix A, we denote its smallest singular value by $\sigma_*(A)$. *i.e.* its invariant factor with smallest norm. An eigenvalue λ of $A \in \mathrm{M}_n(\mathbb{Z}_p)$ is *small* if $|\lambda| < 1$, and *big* otherwise.

1.2. **The iteration subroutine.** Our iteration subroutine is the heart of the main algorithm. In this last section of the introduction, we introduce some definitions to describe the input to the iteration subroutine, and state the results on its output.

Definition 1.2.1. A matrix $\begin{bmatrix} A & * \\ E & B \end{bmatrix} \in \mathrm{M}_n(\mathbb{Z}_p)$ is called *sorted* if for some $\lambda \in \mathbb{Z}_p$ we have

$$E \equiv 0 \pmod{p}, \quad \chi_B(t) \equiv (t-\lambda)^{n_B} \pmod{p}, \quad \text{and } \chi_A(\lambda) \not\equiv 0 \pmod{p}$$

where n_B is the number of columns of the square matrix B. In the special case that $\chi_B(t) \equiv t^{n_B} \pmod{p}$, we say that the matrix is *size-sorted*.

If M is a sorted matrix whose B-block has size 1, then the shift $M - (M\{n,n\})I$ is a size-sorted matrix. A sorted Hessenberg matrix is a matrix which is both sorted and in Hessenberg form. Similarly, a size-sorted Hessenberg matrix is a size-sorted matrix in Hessenberg form. As these matrices feature prominently in our discussion of the QR-algorithm, we give them a special notation.

Definition 1.2.2. We denote by $[A;\epsilon,B]$ a sorted Hessenberg matrix of the form

$$[A;\epsilon,B] := \left[\begin{array}{cc|c} \multicolumn{2}{c|}{A} & * \\ \hline 0 & \epsilon & \multirow{2}{*}{B} \\ 0 & 0 & \end{array}\right], \qquad \text{with } A \in \mathrm{M}_{n_A}(\mathbb{Z}_p),\ B \in \mathrm{M}_{n_B}(\mathbb{Z}_p),\ \epsilon \in \mathbb{Z}_p.$$

The *block sizes* of $[A;\epsilon,B]$ is the tuple (n_A, n_B). If only one of the block sizes is relevant, we use the wildcard character '$*$' to hold the place of the other entry.

Definition 1.2.3. Let $M \in \mathrm{M}_n(\mathbb{Q}_p)$. A *$QR$-round* (with shift μ) is the computation consisting of the following steps applied to M:

1. Compute a QR-factorization $M - \mu I = QR$
2. Set $M_{\text{next}} := RQ + \mu I$

If a value for the shift μ is not mentioned explicitly, we mean $\mu = 0$ by default. It will always be clear from context to which matrix we apply the QR-round steps when we use the term.

To clarify our terminology, the term *QR-iteration* broadly refers to a process consisting of multiple QR-rounds applied to an input matrix, particularly when we do not wish to specify the shifts or the number of rounds for the sake of exposition. Alternatively, Algorithm 4.3, which is titled `QR_Iteration`, is a QR-iteration where the number of QR-rounds is determined in advance based on the input and the shifts are chosen deterministically during the iteration.

We now state our technical result regarding the convergence of the QR-iteration applied to a size-sorted Hessenberg matrix.

Proposition (4.4.1)**.** *Let $M := [A;\epsilon,B]$ be a size-sorted Hessenberg matrix, let $m = n_B$ and let $\lambda_1, \ldots, \lambda_m$ be the small eigenvalues of M. If $\eta := \max_{i,j} |\lambda_i - \lambda_j| \leq |\epsilon|$, then after m QR-rounds we obtain a size-sorted Hessenberg matrix $[A_{\text{next}};\epsilon_{\text{next}},B_{\text{next}}]$ such that $|\epsilon_{\text{next}}| \leq |\epsilon^2|$. Each round uses $2n^2 + o(n^2)$ operations of $\mathbb{Q}_p$ arithmetic. After at most $(m\lceil \log_2(-\log_p \eta)\rceil)$ rounds, the obtained $[A_{\text{next}};\epsilon_{\text{next}},B_{\text{next}}]$ is such that $|\epsilon_{\text{next}}| < \eta$.*

Note that if $\eta \leq p^{-N}$ (which vacuously occurs when $m = 1$), we need at most $(m\lceil \log_2 N\rceil)$ QR-rounds (with shifting) to deflate ϵ to $0 + O(p^N)$.

Remark 1.2.4. If $A + O(p^N)$ is an $n \times n$-matrix whose entries are chosen with the uniform probability distribution on $[0, \ldots, p^N - 1]$, then the limit as $n \to \infty$ of the probability that χ_A is square-free is at least $\frac{1-p^{-5}}{1+p^{-3}}$ [10].

2. Background

2.1. **Precision and QR-factorizations.** We state some basic definitions for our discourse. We follow [8] for terminology, and direct the reader to [3, 5, 11] for more details. We can identify a subgroup of $\mathrm{GL}_n(\mathbb{Q}_p)$ where every matrix is well-conditioned, serving the analogous role to $\mathrm{O}_n(\mathbb{R})$ in the real setting.

Lemma 2.1.1. *Let $A \in \mathrm{GL}_n(\mathbb{Q}_p) \cap \mathrm{M}_n(\mathbb{Z}_p)$. Then the following are equivalent:*
(a) $A \in \mathrm{GL}_n(\mathbb{Z}_p)$
(b) $\|A\| = \left\|A^{-1}\right\| = 1$
(c) The roots of χ_A lie in $\overline{\mathbb{Z}_p}^{\times}$.

Proof. For (a) if and only if (b), it is direct consequence of the fact that $\|A\| = |\sigma_1(A)|$, the first invariant factor. For (c), see [5, Theorem 4.3.8]. □

Proposition/Definition 2.1.2 (p-adic QR-factorization)**.** Let $A \in \mathbb{Z}_p^{n\times m}$ be a matrix. Then there exists a $Q \in \mathrm{GL}_n(\mathbb{Z}_p)$ and an upper triangular matrix $R \in \mathbb{Z}_p^{n\times m}$ such that $A = QR$.

Proof. See [5, Chapter 4], or note this follows from the Iwasawa decomposition of $\mathrm{GL}_n(\mathbb{Q}_p)$. □

For a matrix $M \in \mathrm{M}_n(\mathbb{Z}_p)$, the QR-factorization is generally not unique. For example, if $M := QR$ is a QR-decomposition, and $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ is an upper triangular matrix, We have that $(QU)(U^{-1}R)$ is also an upper triangular matrix. The following type of QR-decomposition is well suited to understand the kernel and rank of a matrix.

Definition 2.1.3. We say that $M = QR$ is a *strict* QR-factorization if for each $i \geq 2$, the first non-zero entry of the i-th row of R is strictly to the right of the first non-zero entry of the $(i-1)$-th row. That is, R is a matrix in echelon form.

Over $\mathbb{Z}_p$, a strict QR-decomposition for M reveals the rank of M as the number of non-zero pivots. Unfortunately, with insufficient precision not all strict QR-forms of a matrix reveal the rank in this way – we discuss this further in Example 2.2.10.

Let $M \in \mathrm{M}_n(\mathbb{Z}_p)$ be a matrix. The *Smith normal form* for M is a diagonal matrix Σ such that the diagonal elements $\sigma_1, \ldots, \sigma_n$ satisfy $|\sigma_1| \geq \ldots \geq |\sigma_n|$ and $M = U\Sigma V$ for some $U, V \in \mathrm{GL}_n(\mathbb{Z}_p)$. The Smith normal form is the pnumerical analogue of the singular value decomposition from standard numerical linear algebra.

Definition 2.1.4. Let $M \in \mathrm{M}_{m\times n}(\mathbb{Z}_p)$ be a matrix and let $M = U\Sigma V$, with Σ the Smith normal form and $U \in \mathrm{GL}_m(\mathbb{Z}_p), V \in \mathrm{GL}_n(\mathbb{Z}_p)$. The *$p$-adic singular value decomposition* of M is the decomposition $M = U\Sigma V$. The *singular values* of M are sizes of the diagonal entries of Σ.

Since we are never concerned with matrices over the reals, we will simply use the terms "QR-decomposition/factorization" or "singular value decomposition" without the p-adic prefix in the sequel.

Remark 2.1.5. With pivots chosen with respect to the p-adic norm, several standard algorithms also work for matrices over $\mathbb{Q}_p$. Specifically:
(a) The standard algorithm to compute the Hessenberg form computes a Hessenberg form [4].
(b) The standard algorithm to compute a PLU-decomposition computes a PLU-decomposition. Moreover, $P^{-1}L \in \mathrm{GL}_n(\mathbb{Z}_p)$, so this is a p-adic QR-decomposition [5, Chapter 4].
(c) One can modify the algorithm in (b) to allow column pivoting, and then factor $U = \Sigma V$ with Σ diagonal and $V \in \mathrm{GL}_n(\mathbb{Z}_p)$ to compute a p-adic singular value decomposition. See the proof of [5, Theorem 4.3.4] for further details.

Remark 2.1.6. If M is a Hessenberg matrix, we can restrict the permutations used in the standard PLU-factorization algorithm to compute a QR-factorization $M = QR$ such that Q is a Hessenberg matrix. Then $M_{\text{next}} := RQ$ is the product of a Hessenberg matrix with an upper-triangular matrix, so is also a Hessenberg matrix. Moreover, at worse $2n$ row operations (n row eliminations plus n row transpositions) are needed to compute Q and R from M. Computing $M_{\text{next}} = RQ$ can then be done in $2n$ columns operations. Since row/column permutations do not require arithmetic operations (only memory allocations or pointer reassignment, depending on the implementation), the cost of one QR-round applied to a Hessenberg matrix is bounded by n^2 arithmetic operations. If we also compute an update $V \mapsto Q^{-1}V$ to a transformation matrix, the total cost is $2n^2$ arithmetic operations.

We now come to the discussion of p-adic precision. There are many ways to represent a p-adic element $a \in \mathbb{Q}_p$ in a computer system [11]. We represent an element of $\mathbb{Q}_p$ by a truncated series

$$a = a_{-r}p^{-r} + \ldots + a_0 + pa_1 + a_2p^2 + \ldots + a_{N-1}p^{N-1} + O(p^N)$$

where the $O(p^N)$ is the p-adic ball representing the uncertainty of the remaining digits. The *relative precision* of a is the quantity $N + r$, and the *absolute precision* is the number N. In the terminology of [11], we consider a system with the *zealous* (i.e, *interval*) implementation of arithmetic. The operations $-, +$ preserve the minimum of the absolute precision of the operands, and $\times, \div$ preserve the minimum relative precision of the operands. If $u \in \mathbb{Z}_p^\times$, $a \in \mathbb{Z}_p$, and $N \leq N'$, then we have that $(u + O(p^{N'}))(a + O(p^N)) = ua + O(p^N)$. Multiplication by p preserves the relative precision and increases the absolute precision by 1. The worst operation when it comes to absolute p-adic precision is dividing a small number by p. For example, the expression

$$\frac{(1 + p^{99} + O(p^{100})) - (1 + O(p^{100}))}{p^{100} + O(p^{200})} = p^{-1} + O(1)$$

begins with 3 numbers with an absolute and relative precision of at least 100, and ends with a result where not even the constant term is known. Henceforth, by *precision* we refer to the absolute precision.

Definition 2.1.7. Let $A, B \in \mathrm{M}_n(\mathbb{Z}_p)$ be matrices such that $a_{i,j} = b_{i,j} + O(p^{N_{i,j}})$. Then we write $A = B + O(p^N)$, where $N := \min_{i,j} N_{i,j}$.

To refer to a matrix $A \in \mathrm{M}_n(\mathbb{Q}_p)$ whose elements are known at an absolute precision at least N, we will simply write $A + O(p^N)$. The same absolute precision on every entry is called a *flat* precision.

2.2. **Orthogonality and the Bilinear Lemma.** In pnumerical linear algebra, we often need to bridge the gap between an approximate computation – usually, where arithmetic is performed in the ring $\mathbb{Z}_p/p^N\mathbb{Z}_p$ – and some information about the true solution to our problem over $\mathbb{Z}_p$. For example, consider computing the kernel of the following matrix equation

$$Mx = \begin{bmatrix} p^3 & 0 \\ 0 & 0 \end{bmatrix} x = 0.$$

Over $\mathbb{Z}_p$, we see that this matrix plainly has rank 1, and our kernel is given by e_2. However, the kernel of $M \otimes_{\mathbb{Z}_p} \mathbb{Z}_p/p^N\mathbb{Z}_p$ will always be rank 2 as a $\mathbb{Z}_p/p^N\mathbb{Z}_p$-module. Thus, it is helpful to understand the properties of $\ker_{\mathbb{Z}_p} M \otimes_{\mathbb{Z}_p} \mathbb{Z}_p/p^N\mathbb{Z}_p$ to best make sense of the approximate computations. This leads us to the concept of p-adic orthogonality as introduced in [9].

Definition 2.2.1. A set $\{x_1, \ldots, x_r\} \subset \mathbb{Q}_p^n$ is *orthogonal* if for every $\lambda_1, \ldots, \lambda_r \in \mathbb{Q}_p$ we have that

$$\left\| \sum_{j=1}^{r} \lambda_j x_j \right\| = \max \{ |\lambda_j| \, \|x_j\| : 1 \leq j \leq r \}.$$

We say $\{x_1, \ldots, x_r\}$ is *orthonormal* if it is orthogonal and each $\|x_j\| = 1$.

Definition 2.2.2. A submodule $V \subseteq \mathbb{Z}_p^n$ is *orthonormally generated* if it is generated by an orthonormal set. We also say that V *admits an orthogonal basis*.

Note that a subset $\{x_1, \ldots, x_r\} \subset \mathbb{Z}_p^n$ is orthonormal if and only if $r \leq n$ and there is a $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ such that $x_j = e_j U$ for all $1 \leq j \leq r$. Since any two bases of a free $\mathbb{Z}_p$-module are related by a transformation in $\mathrm{GL}_n(\mathbb{Z}_p)$, we obtain the following basis-free characterizations of the orthonormally generated criterion.

Lemma 2.2.3. *Let V be a free $\mathbb{Z}_p$-submodule of $\mathbb{Z}_p^n$.*

(a) If V admits an orthonormal basis, then every basis of V is orthonormal.
(b) We have that V is orthonormally generated if and only if the cokernel of the inclusion $V \hookrightarrow \mathbb{Z}_p^n$ is a free $\mathbb{Z}_p$-module.

We additionally have a notion of orthogonal complement.

Definition 2.2.4. Two submodules $U, V \subseteq \mathbb{Z}_p^n$ are orthogonal if for some choice of bases $\{u_i, i \in I\}$, $\{v_j, j \in J\}$ the set $\{u_i, i \in I\} \cup \{v_j, j \in J\}$ is orthogonal. If V and U are both orthonormally generated and $\mathbb{Z}_p^n = V \oplus U$, we say that U is *an orthogonal complement* to V (and *vice-versa*).

Given a submodule $V \subseteq \mathbb{Z}_p^n$ that is orthonormally generated, it is easy to construct an orthogonal complement. Writing a basis for V as the rows of an $r \times n$ matrix M, we compute a singular value decomposition $M = Q\Sigma P$. Note that Σ has unit entries on the diagonal, as V is orthonormally generated, and that the first r rows of P generate V as a submodule. Since $P \in \mathrm{GL}_n(\mathbb{Z}_p)$, we see that the last $n - r$ rows of P generate an orthonormal module orthogonal to V. That being said, the orthogonal complement of a non-trivial subspace is never unique.

A useful result to relate the results of our computations back to results over $\mathbb{Z}_p$ is the Bilinear Lemma of Samuel-Zariski [12, Chapter VIII, Section 7].

Lemma 2.2.5 (Bilinear Lemma)**.** *Let A be a ring, $\mathfrak{m}$ an ideal in A, and let E, E', F be three A-modules. Assume that F is a Hausdorff space for its $\mathfrak{m}$-topology and that A is complete. Let $f\colon E \times E' \to F$ be a bilinear mapping, and denote by $\bar{f}\colon E/\mathfrak{m}E \times E'/\mathfrak{m}E' \to F/\mathfrak{m}F$ the canonically determined map.*

If we are given $y \in F, \bar{\alpha} \in E/\mathfrak{m}E, \bar{\alpha}' \in E'/\mathfrak{m}E'$ such that $\bar{f}(\alpha, \alpha') = \bar{y}$ and $F/\mathfrak{m}F = f(\bar{\alpha}, E'/\mathfrak{m}E') + f(E/\mathfrak{m}E, \bar{\alpha}')$. Then there are lifts of α, α' to E, E' such that $y = f(\alpha, \alpha')$.

We can translate this directly to our situation.

Lemma 2.2.6 (Bilinear Lemma, specialized)**.** *Let $\mathfrak{m}$ an ideal in $\mathbb{Z}_p$. If we are given $y \in \mathbb{Z}_p^n, \bar{x} \in (\mathbb{Z}_p/\mathfrak{m}\mathbb{Z}_p)^n, \bar{M} \in \mathrm{M}_n(\mathbb{Z}_p/\mathfrak{m}\mathbb{Z}_p)$ such that $\bar{x}$ has a unit coordinate and $\bar{x}\bar{M} = \bar{y}$. Then there is a lift $x \in \mathbb{Z}_p^n$ of $\bar{x}$ and a lift $M \in \mathrm{M}_n(\mathbb{Z}_p)$ of $\bar{M}$ such that $xM = y$.*

Proof. The hypotheses of the general Bilinear Lemma are readily checked. □

Finally, we define the notion of orthogonality, orthonormal, and orthogonal complement for $(\mathbb{Z}_p/p^N\mathbb{Z}_p)^n$.

Definition 2.2.7. Let V be a submodule of $(\mathbb{Z}_p/p^N\mathbb{Z}_p)^n$. Then V is *orthonormally generated* if the cokernel of the inclusion $V \hookrightarrow (\mathbb{Z}_p/p^N\mathbb{Z}_p)^n$ is a free $(\mathbb{Z}_p/p^N\mathbb{Z}_p)$-module.

Definition 2.2.8. Two submodules $U, V \subseteq (\mathbb{Z}_p/p^N\mathbb{Z}_p)^n$ are orthogonal if for some choice of bases $\{\bar{u}_i, i \in I\}$, $\{\bar{v}_j, j \in J\}$ and any lifts $\{u_i, i \in I\}$, $\{v_j, j \in J\}$ to $\mathbb{Z}_p^n$, the set $\{u_i, i \in I\} \cup \{v_j, j \in J\}$ is orthogonal. If V and U are both orthonormally generated and $(\mathbb{Z}_p/p^N\mathbb{Z}_p)^n = V \oplus U$, we say that U is *an orthogonal complement* to V (and *vice-versa*).

2.2.1. pNumerical ranks, kernels, and preimages

In this section, we define the pnumerical rank, kernel, and inverse image. We also discuss how to compute such objects and how they relate to their exact counterparts for a matrix $M \in \mathrm{M}_n(\mathbb{Z}_p)$.

Definition 2.2.9. The *pnumerical rank of precision $O(p^N)$* of $M \in \mathrm{M}_n(\mathbb{Z}_p)$ is the number of singular values of M of norm strictly bigger than p^{-N} (*i.e.* of valuation strictly smaller than N).

With a sufficient amount of precision, the pnumerical rank will be equal to the rank. Additionally, the strict QR-factorization will reveal the pnumerical rank of the original matrix as the number of non-zero pivots of R. If not enough precision is given, this cannot be guaranteed. The singular value decomposition always reveals the pnumerical rank.

Example 2.2.10. For the matrix

$$M := \begin{bmatrix} p & 1 & \\ 0 & p & 1 \\ 0 & 0 & p \end{bmatrix} + O(p^3)$$

we see with $Q := 1, R := M$ that $M = QR$ is a strict QR-factorization. However, because of the low precision ($|\sigma_*(M)| \leq |p^3|$), we can obtain another strict QR-factorization with

$$Q' := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -p^2 & p & 1 \end{bmatrix} + O(p^3), \qquad R' := \begin{bmatrix} p & 1 & \\ 0 & p & 1 \\ 0 & 0 & 0 \end{bmatrix} + O(p^3).$$

We see that the second strict QR-factorization reveals the pnumerical rank, and the first does not.

We now discuss pnumerical kernels and pnumerical inverse images.

Definition 2.2.11. Let $N > 0$. The *pnumerical kernel of precision* $O(p^N)$ of $M \in \mathrm{M}_n(\mathbb{Z}_p)$ is the maximal free $(\mathbb{Z}_p/p^N\mathbb{Z}_p)$-submodule of $(\mathbb{Z}_p/p^N\mathbb{Z}_p)^n$ annihilated by M.

Definition 2.2.12. Let $N > 0$. The *pnumerical preimage of precision* $O(p^N)$ of a submodule $V \subseteq (\mathbb{Z}_p/p^N\mathbb{Z}_p)^n$ under $M \in \mathrm{M}_n(\mathbb{Z}_p)$ is the the maximal free $(\mathbb{Z}_p/p^N\mathbb{Z}_p)$-submodule $U \subseteq (\mathbb{Z}_p/p^N\mathbb{Z}_p)^n$ such that $MU \subseteq V$.

The pnumerical kernel of M is not generally the kernel of $M \pmod{p^N}$ as an endomorphism of $(\mathbb{Z}_p/p^N\mathbb{Z}_p)^n$. As expected, the pnumerical kernel is just the pnumerical preimage of 0. Generally, if there is no risk of confusion we will forgo stating the "of precision $O(p^N)$" part of these terms.

Lemma 2.2.13. *Let V be a submodule of $(\mathbb{Z}_p/p^N\mathbb{Z}_p)^n$. Then there exists a matrix $M \in \mathrm{M}_n(\mathbb{Z}_p/p^N\mathbb{Z}_p)$ such that the kernel of M (as an endomorphism of $(\mathbb{Z}_p/p^N\mathbb{Z}_p)^n$) is exactly V. If V is orthonormally generated of rank b, then M has $n-b$ singular values of size 1 and b singular values of size 0.*

Proof. Note that V is a finitely generated $\mathbb{Z}_p$-module, so by the structure theorem for modules over a PID we have that there is an isomorphism $\varphi\colon V \longrightarrow \bigoplus_j (\mathbb{Z}_p/p^j\mathbb{Z}_p)^{m_j}$ with all but finitely many $m_j \in \mathbb{N} \cup \{0\}$ equal to 0. We let B be the finite subset of V obtained by pulling back a set of generators for the direct summands of $\bigoplus_j (\mathbb{Z}_p/p^j\mathbb{Z}_p)^{m_j}$ under φ. Denote $b := \#B$.

Let $X \in \mathrm{M}_n(\mathbb{Z}_p/p^N\mathbb{Z}_p)$ be the $n \times b$ matrix whose columns are the elements of B, and let $X = Q\Sigma P$ be a singular value decomposition. In particular, we have that the bottom $(n-b) \times b$ block of Σ is 0. We lift the entries of Σ to $\mathbb{Z}_p$ and construct the $n \times n$ matrix M by

$$M := \begin{bmatrix} p^N\sigma_1^{-1} & & & & & \\ & \ddots & & & & \\ & & p^N\sigma_b^{-1} & & & \\ & & & 1 & & \\ & & & & \ddots & \\ & & & & & 1 \end{bmatrix} Q^{-1} \in \mathrm{M}_n(\mathbb{Z}_p/p^N\mathbb{Z}_p).$$

We see $MX = 0$. Thus, the kernel of M is exactly V, so this completes the first part of the lemma. If V is orthonormally generated, then our b is also the rank of V and each of the $\sigma_j = 1$, so the second part follows. □

Proposition 2.2.14. *Let $N \in \mathbb{N}$ and let $M \in \mathrm{M}_n(\mathbb{Z}_p)$ be a matrix given at flat precision $O(p^N)$ and let $V \subseteq \mathbb{Z}_p^n$ be a $\mathbb{Z}_p$-submodule. Then:*

(a) The pnumerical kernel of M is orthonormally generated. Furthermore, the pnumerical kernel contains the image of $\ker M$ under reduction modulo p^N.

(b) If V is orthonormally generated, the pnumerical preimage of V is orthonormally generated. Furthermore, the pnumerical preimage of V contains the image of the preimage of V under reduction modulo p^N.

Proof. Both parts can be deduced by using the p-adic singular value decomposition (i.e, the Smith normal form). For part (a), we see the result from the singular value decomposition for M. For part (b), we apply Lemma 2.2.13 to find a matrix A such that $\ker A = V$. In particular, the pnumerical inverse image of V under M is the pnumerical kernel of AM, so we can compute it via the singular value decomposition as before. The statements regarding reduction modulo p^N are obvious. □

Proposition 2.2.14 is optimal in the following sense. If M is known at flat precision $O(p^N)$ and $\bar{V}$ is the pnumerical kernel of M, then there exists some $M' \in \mathrm{M}_n(\mathbb{Z}_p)$ and $V \subseteq \mathbb{Z}_p^n$ such that $M'V = 0$, $M = M' + O(p^N)$, and $\bar{V} = V \otimes \mathbb{Z}_p/p^N\mathbb{Z}_p$. This is an immediate consequence of the Bilinear Lemma. The analogous statement is true for the pnumerical inverse image.

2.3. **The basic QR-algorithm.** Algorithm 2.1 below is the simple QR-algorithm given in [8] (Algorithm 2.19 *loc. cit.*). This version suffers from a number of drawbacks: the algorithm only converges linearly and cannot decompose any block with eigenvalues that are the same modulo p. The core idea to improve the algorithm is the classic strategy of concurrently updating the approximation to the eigenvalue and the matrix.

Algorithm 2.1 `simple_QR_Iteration`$(M, \chi_{A,p})$

Input:
$M + O(p^N)$, an $n \times n$-matrix $in\, \mathrm{M}_n(\mathbb{Z}_p)$.
$\chi_{M,p}$, the characteristic polynomial of $M \pmod p$.

Output: A (block) triangular form T for M, and a matrix V such that $MV = VT + O(p^N)$ (*i.e.* V is a change of basis matrix between T and M).

1: Set $\lambda_1, \ldots, \lambda_\ell$ to be the roots of $\chi_{M,p}$ in $\mathbb{F}_p$, lifted to $\mathbb{Z}_p$.
2: Set $m_1, \ldots, m_\ell$ to be the multiplicities of the roots of $\chi_{A,p}$.
3: Compute B, V such that $MV = VB$ and B is in Hessenberg form.
4: **for** $i = 1, \ldots, \ell$ **do**
5: **for** $j = 1, \ldots, m_i N$ **do**
6: Factor $(B - \lambda_i I) = QR$
7: Set $B := RQ + \lambda_i I$
8: Set $V := Q^{-1}V$
9: **return** B, V

We point out a useful lemma of Wilkinson from [13], which helps in analysing the diagonal elements of the various upper triangular factors encountered in the iteration.

Lemma 2.3.1 (Wilkinson)**.** *Let $M \in \mathrm{M}_n(\mathbb{Q}_p)$ be a matrix, let $s \geq 1$ be an integer, and let $(Q^{(1)}, R^{(1)})$, …, $(Q^{(s)}, R^{(s)})$ be the QR-pairs for s QR-rounds. Let*

$$M^{(s)} := R^{(s-1)}Q^{(s-1)}, \quad \mathcal{Q}^{(s)} := Q^{(1)} \cdots Q^{(s)}, \quad \mathcal{R}^{(s)} := R^{(s)} \cdots R^{(1)}.$$

Then $M^s = \mathcal{Q}^{(s)}\mathcal{R}^{(s)}$ and $M^{(s+1)} = {\mathcal{Q}^{(s)}}^{-1} M \mathcal{Q}^{(s)}$.

We quote from [13, Section 5] a brief summary of Wilkinson's argument to show why QR-iteration converges, in a simple case. We refer to Wilkinson's original article for the other cases. Let $M \in \mathrm{M}_n(\mathbb{Z}_p)$ and assume that $M = XDX^{-1} = XDY$ for D a diagonal matrix with $\lambda_j := D\{j,j\}$ and $|\lambda_1| > \ldots > |\lambda_n| > 0$ and let $X \in \mathrm{GL}_n(\mathbb{Z}_p)$. We will further assume that $X = L_X R_X$ and $Y = L_Y R_Y$ for some $L_X, L_Y \in \mathrm{GL}_n(\mathbb{Z}_p)$ unit lower-triangular matrices and $R_X, R_Y \in \mathrm{GL}_n(\mathbb{Z}_p)$ upper-triangular.

Letting $(L^{(1)}, R^{(1)})$, …, $(L^{(s)}, R^{(s)})$ be the QR-pairs for s QR-rounds (with $M = L^{(1)}R^{(1)}$), we have

$$M^s = XD^sY = X(D^sL_YD^{-s})(D^sR_Y).$$

We write $D^sL_YD^{-s} = I + F_s$, and we have

$$F_s\{i,j\} = \begin{cases} L_Y\{i,j\} \cdot \left(\frac{\lambda_i}{\lambda_j}\right)^s & \text{if } i > j \\ 0 & \text{if } i \leq j. \end{cases}$$

By the assumption on the norms of the λ_j's we have that $\lim_{s\to\infty} F_s \to 0$. We have

$$\begin{aligned} XD^sY &= L_XR_X(1 + F_s)D^sR_Y \\ &= L_X(I + R_XF_sR_X^{-1})R_XD^sR_Y. \end{aligned}$$

Since $F_s \to 0$ under the iteration, for some sufficiently large s we have that the QR-factorization of $(I + R_XF_sR_X^{-1})$ is of the form $(I + L')(I + R')$ with L', R' lower (resp. upper) triangular and tending to 0. In particular, by Wilkinson's Lemma

$$\mathcal{L}^{(s)}\mathcal{R}^{(s)} = \underbrace{L_X(I + L')}\,\underbrace{(I + R')(R_XD^sR_Y)}.$$

The left factor is lower triangular and the right factor is upper triangular, so by the uniqueness of LR-decompositions of non-singular matrices over a domain, we have that $\mathcal{L}^{(s)} = L_X(I + L')$. But now

with $M^{(s)}$ the s-th iterate of M under the QR-iteration we have by Wilkinson's Lemma

$$\begin{aligned} M^{(s)} &= (\mathcal{L}^{(s)})^{-1} M (\mathcal{L}^{(s)}) \\ &= (\mathcal{L}^{(s)})^{-1} X D X^{-1} (\mathcal{L}^{(s)}) \\ &= (I + L')^{-1} L_X^{-1} L_X R_X D R_X^{-1} L_X^{-1} L_X (I + L') \\ &= (I + L')^{-1} R_X D R_X^{-1} (I + L'). \end{aligned}$$

Because $\lim_{s\to\infty}(I + L') = I$, we see the $M^{(s)}$ converge to the upper triangular matrix $R_X D R_X^{-1}$.

2.4. **Problematic examples.** Before proceeding with the rest of the article, we include examples that highlight some of the technical difficulties we need to be aware of in our proofs. First, we review an example from [8].

Example 2.4.1. Consider the matrix

$$A := \begin{bmatrix} p^3 & p^2 \\ 0 & -p^3 \end{bmatrix} + O(p^6).$$

The characteristic polynomial computed using capped precision arithmetic is $\chi_A + O(p^6) = T^2 + O(p^6)$. There is a precision loss in computing the roots of f, and the absolute error on the roots of f cannot be better than $O(p^3)$. However, it is possible to know the characteristic polynomial of A at a higher precision; keeping track of extra digits of precision, we have

$$\begin{aligned} \chi_A &= (p^3 + O(p^6) - T)(-p^3 + O(p^6) - T) - (p^2 + O(p^6))(0 + O(p^6)) \\ &= T^2 - (p^3 - p^3 + O(p^6))T + (p^6 + O(p^9)) - (O(p^8)) \\ &= T^2 - (0 + O(p^6))T + (p^6 + O(p^8)). \end{aligned}$$

With the extra digits of precision on the last coefficient of χ_A, we can compute the roots of χ_A with an absolute error of $O(p^4)$. In particular, even when the input has flat precision, there are cases where the characteristic polynomial needs to be known at higher precision to obtain the best accuracy on the eigenpairs.

Next, we discuss topologically nilpotent matrices.

Definition 2.4.2. We say a matrix $M \in \mathrm{M}_n(\mathbb{Z}_p)$ is *topologically nilpotent* if $\lim_{j\to\infty} \left\|M^j\right\| = 0$.

For example, any matrix of the form

$$\begin{bmatrix} 0 & \dots & & 0 \\ 1 & & & \\ & \ddots & & \\ & & 1 & 0 \end{bmatrix} + pX, \quad X \in \mathrm{M}_n(\mathbb{Z}_p)$$

is topologically nilpotent. Topologically nilpotent matrices generally exhibit the worst-case scenario for the computation of the characteristic polynomial or iterative eigenvector algorithms [4, 8]. Practically, either more precision or more iterations are required to compute the generalized eigenspaces in these cases. Topologically nilpotent matrices are a particular examples of matrices $M \in \mathrm{M}_n(\mathbb{Q}_p)$ such that $|\lambda_1 - \lambda_2| < 1$ for some eigenvalues λ_1, λ_2 of M. In the archimedean case, the distances between eigenvalues of an input matrix is well-known to be related to the condition number of the eigenvalue/eigenvector problem.

Example 2.4.3 (Topologically nilpotent matrices)**.** Topologically nilpotent blocks present a worst case scenario for the convergence of our QR method. Consider the $(n+1) \times (n+1)$ matrix

$$\begin{bmatrix} 1 & & & & \\ p & 0 & \dots & 0 & p \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & 0 \end{bmatrix} + O(p^N)$$

After $n-1$ rounds (resp. n) QR-rounds (with shift 0), we end up with the matrix

$$\begin{bmatrix} 1 & & & & & \\ p & 0 & \dots & & 0 & 1 \\ & p & & & & \\ & & 1 & & & \\ & & & \ddots & & \\ & & & & 1 & 0 \end{bmatrix} + O(p^N), \qquad (resp.) \qquad \begin{bmatrix} 1 & & & & & \\ p^2 & 0 & \dots & & 0 & p \\ & 1 & & & & \\ & & 1 & & & \\ & & & \ddots & & \\ & & & & 1 & 0 \end{bmatrix} + O(p^N).$$

We see that the convergence of the $(2,1)$-entry to zero is hampered by the chain of subdiagonal 1's. If λ_1, λ_2 are distinct small eigenvalues and $\gcd(p,n) = 1$, we have $|\lambda_1 - \lambda_2| = p^{-\frac{1}{n}}$, so we do not meet the criterion for quadratic convergence. Second, this example suggests even in optimal cases why we may need $n \log_2 N$ iterations for the $(2,1)$ entry to converge to zero modulo p^N; essentially, we can only guarantee that the size of this entry decreases within n iterations.

Example 2.4.4 (Disordered eigenvalues)**.** Consider the matrix

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 3 & 0 & 0 \\ & p & p & p & 1 \\ & & p^{10} & 1 & 2 \\ & & & p & p \end{bmatrix} + O(p^N).$$

It is not immediately clear what the change of coordinates is to ensure that the matrix remains in Hessenberg form and for the backward orbit of 0 (mod p) to correspond to the last two (row) vectors. The transformation to convert this matrix to a size-sorted Hessenberg matrix appears to be difficult to compute.

There are several ways in which a matrix in $\mathrm{M}_n(\mathbb{Z}_p)$ can fail to be diagonalized by a $\mathrm{GL}_n(\mathbb{Z}_p)$ transformation. The first is that the matrix is not semi-simple, and the second is that the characteristic polynomial of M may contain non-trivial irreducible factors. There is a third obstruction to diagonalizability whenever the singular values differ from the sizes of the eigenvalues.

Example 2.4.5 (Non-$\mathrm{GL}_n(\mathbb{Z}_p)$-diagonalizable matrices.)**.** Consider the topologically nilpotent matrix

$$M = \begin{bmatrix} p & 1 \\ 0 & 0 \end{bmatrix}.$$

We see that M is in Schur form and that the eigenvalues are $\{0, p\}$. It is impossible to diagonalize M over $\mathrm{GL}_n(\mathbb{Z}_p)$, as $\mathrm{GL}_n(\mathbb{Z}_p)$ conjugation preserves the singular values, which in this case are $\{0, 1\}$.

3. Computing size-sorted forms and generalized 0-eigenspaces

In this section, we study the connection between size-sorted forms of a matrix and approximations to the generalized 0-eigenspace. We first present a refinement of the Hodge-Newton decomposition from [5, Theorem 4.3.11]. To begin, we give a variant of a classical result.

Lemma 3.0.1. *Let $f \in \mathbb{Z}_p[t]$ be a polynomial, let $M \in \mathrm{M}_n(\mathbb{Z}_p)$, let $V := \operatorname{lker} f(M)$, and let $r := \operatorname{rank} V$. Then V is orthonormally generated. Moreover, there exists a $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ such that $VU^{-1} = \langle e_{n-r+1}, \dots, e_n \rangle$. In particular,*

$$UMU^{-1} = \begin{bmatrix} A & C \\ 0 & B \end{bmatrix}.$$

Proof. Since V is a kernel, it is orthonormally generated and admits an orthogonal complement $V^\perp$. Representing a basis $b_1, \dots, b_{n-r}$ for $V^\perp$ and a basis $b_{n-r+1}, \dots, b_n$ for V as row vectors we construct

$$U := \begin{bmatrix} b_1^T & \cdots & b_n^T \end{bmatrix}^T.$$

By orthogonality we have $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ and by definition U sends $\langle e_{n-r+1}, \dots, e_n \rangle$ to V. Finally, M commutes with $f(M)$, so V is an invariant subspace for M. In particular, $VM \subseteq V$. By the definition of U we have $UMU^{-1} = \begin{bmatrix} A & C \\ 0 & B \end{bmatrix}$ as required. □

The lemma above allows us to show that a factorization of χ_M indicates that M can be put into a matching block triangular form by a $\mathrm{GL}_n(\mathbb{Z}_p)$ transformation. We can now prove the first theorem from the introduction.

Theorem 3.0.2. *Let $M \in \mathrm{M}_n(\mathbb{Z}_p)$ and let $\chi_M = f_1 \cdots f_r$ be a factorization in $\mathbb{Z}_p[t]$ where the factors are pairwise coprime in $\mathbb{Q}_p[t]$. Then there exists a $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ such that UMU^{-1} is block-triangular with r blocks; the j-th block accounts for the eigenvalues λ such that $f_j(\lambda) = 0$.*

Proof. Using Lemma 3.0.1 with the polynomial f_r, we can find a $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ such that

$$UMU^{-1} = \begin{bmatrix} A & C \\ 0 & B \end{bmatrix}.$$

Since the f_j are pairwise coprime, we have $\chi_B = f_r$. The result follows from an inductive argument. □

Even though a $\mathrm{GL}_n(\mathbb{Z}_p)$ transform can be found to put a matrix into a block Schur form, this does not mean a $\mathrm{GL}_n(\mathbb{Z}_p)$ matrix can be found that block diagonalizes the matrix. See Example 2.4.5. If $f \in \mathbb{Z}_p[t]$ is a polynomial whose roots have valuations $\{\nu_1, \ldots, \nu_R\}$, there is a factorization $f = f_1 \cdots f_r$ where the roots of each f_j have valuation ν_j (see [5, Section 2.2], [14–16] for more details on the factorization of p-adic polynomials, slope factorization and how to compute them). Thus we obtain:

Corollary 3.0.3 (Newton decomposition)**.** *Let $M \in \mathrm{M}_n(\mathbb{Z}_p)$ and let $\nu_1 \leq \ldots \leq \nu_r$ be the distinct valuations of the eigenvalues of M. Then there exists a $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ such that UMU^{-1} is block-triangular with r diagonal blocks; the j-th block accounts exactly for the eigenvalues of valuation ν_j.*

To compute a sorted matrix, we can use the standard algorithm to compute a block Schur form for a matrix over $\mathbb{F}_p$. We state this as Algorithm 3.1. Note that a size-sorted form is a 1-digit of precision approximation to the Newton decomposition from Corollary 3.0.3.

Algorithm 3.1 `sorted_form`

Input:
An $n \times n$ matrix M known at precision $O(p^N)$.
Output: A sorted form M' for M and a matrix $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ such that $M' = UMU^{-1}$.

1: Set $\bar{M} := M \pmod p$.
2: Compute a block Schur form for $\bar{M}$ with change of basis matrix $U \in \mathrm{GL}_n(\mathbb{F}_p)$
3: Lift U to $\mathrm{GL}_n(\mathbb{Z}_p)$
4: Set $M' := UMU^{-1}$
5: **return** M', U

This algorithm is sufficient for our purpose of computing the block Schur form. We also see that there is a connection between computing the generalized 0-eigenspace at 1 digit of precision and the computation of a size-sorted form of a matrix. Consequently, sorted matrices necessarily have a non-trivial factorization of their characteristic polynomials.

Lemma 3.0.4. *Let $M := \begin{bmatrix} A & C \\ E & B \end{bmatrix}$ be a size-sorted matrix and let ϵ be a positive power of p such that $E \equiv 0 \pmod \epsilon$. Then there is a factorization $\chi_M = \chi_{\mathrm{big}}\chi_{\mathrm{small}}$ in $\mathbb{Z}_p[t]$ such that $\chi_{\mathrm{big}} \equiv \chi_A \pmod \epsilon$ and $\chi_{\mathrm{small}} \equiv \chi_B \pmod \epsilon$. Moreover, the factorization $\chi_M \equiv \chi_A\chi_B \pmod \epsilon$ into monic polynomials is unique in $(\mathbb{Z}_p/\epsilon\mathbb{Z}_p)[t]$.*

Proof. Note that $\chi_M \equiv \chi_A\chi_B \pmod \epsilon$, and in particular $\chi_M \equiv \chi_A\chi_B \pmod p$. Writing $\chi_{A,p}, \chi_{B,p}$ for the reductions of χ_A, χ_B modulo p (respectively), we have $\gcd(\chi_{A,p}, \chi_{B,p}) = 1 \in \mathbb{Z}_p/p\mathbb{Z}_p$. By Hensel's lemma, we have the factorization $\chi_M = \chi_{\mathrm{big}} \cdot \chi_{\mathrm{small}}$ in $\mathbb{Z}_p[t]$, and moreover, if $\chi_M \equiv FG \pmod \epsilon$ is a factorization such that $F \equiv \chi_{\mathrm{big}} \pmod p$ and $G \equiv \chi_{\mathrm{small}} \pmod p$, then $F \equiv \chi_{\mathrm{big}} \pmod \epsilon$ and $G \equiv \chi_{\mathrm{small}} \pmod \epsilon$. Thus, we see that $\chi_{\mathrm{small}} \equiv \chi_B \pmod \epsilon$. □

The remainder of this section is devoted to computing the generalized 0-eigenspace of a matrix. This offers two possible benefits. First, it allows us to repair the classical algorithm to deal with cases such as Example 3.1.1. Secondly, we can potentially set up the iterative algorithms to block-triangularize topologically nilpotent matrices.

3.1. **Computing the generalized 0-eigenspace: Problematic examples.** In this section we give some examples that demonstrate the difficulty of computing the generalized 0-eigenspace.

Example 3.1.1. Let $A := \begin{bmatrix} p^2 & & \\ & 0 & 1 \\ & 0 & 0 \end{bmatrix} + O(p^4)$. The schoolbook method to compute the (right sided) generalized 0-eigenspace is to compute $\ker A^2$. Unfortunately, we see that $A^2 \equiv 0 \pmod{p^4}$, and in this case we do not compute the generalized 0-eigenspace correctly.

Example 3.1.2. In infinite precision, another way to compute the generalized 0-eigenspace is to iteratively solve $Ax = b$, starting with $b = 0$. The corresponding calculation in finite precision is a little delicate. Consider the matrix

$$A := \begin{bmatrix} p & & & \\ & 0 & 0 & 0 \\ & 1 & 0 & 0 \\ & 0 & p & 0 \end{bmatrix} \in \mathrm{M}_4(\mathbb{Z}_p).$$

We see that the right kernel of A is generated by e_4, and that the generalized 0-eigenspace is $\langle e_2, e_3, e_4 \rangle$. Unfortunately, in this case, the solutions in $\mathbb{Z}_p^4$ to $Ax = e_4$ are of the form $x = p^{-1}e_3 + ue_4$, where $u \in \mathbb{Z}_p$. Working with 4-digits of precision, the $\mathbb{Z}_p/p^4\mathbb{Z}_p$-submodule of elements such that $\bar{A}x \in \langle \bar{e}_4 \rangle$ is $\langle p^3\bar{e}_1, \bar{e}_3, \bar{e}_4 \rangle$. This example indicates we need to be careful about what we mean by the "backward orbit of 0 $(\mathrm{mod}\ p^N)$" and motivates the definition of the *pnumerical preimage of precision* $O(p^N)$ in Definition 2.2.12.

3.2. **The generalized 0-eigenspace algorithm.** We give an algorithm (Algorithm 3.2) to compute the generalized 0-eigenspace of a matrix M given at finite precision.

Algorithm 3.2 Generalized 0-eigenspace (abbreviated to `GZE`)

Input:
An $n \times n$ matrix M known at precision $O(p^N)$.
An s.v.d factorization $M = Q\Sigma P$.

Output: A matrix whose rows form a basis of a numerical approximation of the generalized left 0-eigenspace of M.

1: Set $K := [e_{r+1}, \ldots, e_n] \in \mathbb{Z}_p^{(n-r)\times n}$ with r the pnumerical rank of M at precision $O(p^N)$ to be a matrix representing the pnumerical left kernel of Σ
2: **if** $K = \langle 0 \rangle$ **then**
3: **return** $\emptyset$
4: Set $V := KQ^{-1}$. Set $\delta := n - r$
5: Set B to be a $\delta \times \delta$ square sub-block of V such that $B \in \mathrm{GL}_\delta(\mathbb{Z}_p)$
6: Set $W := B^{-1}V$
7: Set J to be the set of the indices of the pivot columns in W.
8: Eliminate columns of M using pivots from W: call this M'
i.e, compute $X \in \mathrm{M}_n(\mathbb{Z}_p)$ such that for any $j \in J$, we have $(M - XW)\{\bullet, j\} = 0$
9: Delete the columns and rows in M' indexed by J: call this M''
10: Compute $M'' = Q''\Sigma''P''$ an s.v.d. decomposition
11: Set $V_{new} := \mathtt{GZE}(M'' = Q''\Sigma''P'')$
12: Set $\widetilde{V_{new}}$, obtained from V_{new} as a matrix with n columns, with those of index in J being 0
13: **return** $\begin{bmatrix} \widetilde{V_{new}} \\ V \end{bmatrix}$

The underlying reason that this algorithm computes the correct answer is that after the truncation in step 9, we have that V is the kernel of M and that we constructed the operator $\overline{M} \colon \mathbb{Z}_p^n/V \to \mathbb{Z}_p^n/V$ up to a change of basis. We then use the fact that $\mathtt{GZE}(M) \cong \mathtt{GZE}(\overline{M}) \oplus V$. More precisely, one chooses an orthogonal complement $V^\perp$ to V inside $\mathbb{Z}_p^n$ and computes an operator M' such that M' stabilizes $V^\perp$ and the image of $M - M'$ is contained in V. In this case, $\mathtt{GZE}(M) = \mathtt{GZE}(M'|_{V^\perp}^{V^\perp}) \oplus V$. Note that

an orthogonal complement to V in $\mathbb{Z}_p$ is given by $\langle e_i : i \notin J\rangle$. This is easily seen from the fact that the pivots of W occur in the columns indexed by J.

Lemma 3.2.1. *The matrix M'' computed on step 9 represents $\overline{M}\colon \mathbb{Z}_p^n/V \to \mathbb{Z}_p^n/V$.*

Proof. First, we fix the basis $\langle e_i : i \notin J\rangle$ for the choice of orthogonal complement. Note that we have the equation $M' := M - XW$. The (left) image of M' is contained in $\langle e_i : i \notin J\rangle$. In particular, M' defines an endomorphism of the subspace $\langle e_i : i \notin J\rangle$. The explicit matrix describing this endomorphism on $\mathbb{Z}_p^n/V$ with respect to the chosen basis is obtained from M' by deleting the columns indexed by J. This is exactly the matrix M''. □

Proposition 3.2.2. *Let $M \in \mathrm{M}_n(\mathbb{Z}_p)$ be a matrix given at flat precision $O(p^N)$ and let V_0 be the generalized left 0-eigenspace of M. Then Algorithm 3.2 computes an approximation at precision $O(p^N)$ of V_0, in $O(n^3 \dim(V_0))$ arithmetic operations at precision $O(p^N)$.*

Proof. From the previous lemma and discussion, it is clear that Algorithm 3.2 is correct when performing computations at infinite precision. Next, note that the rows of the matrix V computed in step 4 are orthonormal, as the rows of V generate the kernel of M as a morphism of $\mathbb{Z}_p$-modules. Consequently, no divisions by p are needed to compute the reduced row echelon form of V. In the elimination on step 8, the pivot entries of W are units, so no divisions by p are needed to perform the eliminations. Since J indexes both the pivots of V and a collection of identically 0 columns in $\widetilde{V_{new}}$, we see that the rows of V and $\widetilde{V_{new}}$ are orthonormal.

Let $\widetilde{V} := \begin{bmatrix} \widetilde{V_{new}} \\ V \end{bmatrix}$ and let $\delta := \dim(V_0)$. By definition, we see that $\widetilde{V}M^\delta = 0 + O(p^N)$, so the module generated by the rows of $\widetilde{V} \pmod{p^N}$ is contained in the pnumerical kernel of M^δ. On the other hand, the rows of $\widetilde{V}$ are orthonormal, so it is easy to see by the exit condition in step 2 that the rows of $\widetilde{V} \pmod{p^N}$ generate the pnumerical kernel of M^δ. By Proposition 2.2.14 we see that the rows of $\widetilde{V}$ generate $V_0 \pmod{p^N}$.

Finally, we comment on the computational complexity. The s.v.d computation and the eliminations in step 8 can both be done with $O(n^3)$ arithmetic operations. Steps 5 and 6 can be combined and done with $O(n^3)$ arithmetic operations using the QR-decomposition with column pivoting. The number of recursive calls is at most $\dim V_0$. In total, we perform $O(n^3 \dim(V_0))$ arithmetic operations. □

Remark 3.2.3. The repeated computation of the singular value decomposition in Algorithm 3.2 means it is not efficient. Instead of using an s.v.d. decomposition, we can use a QR-decomposition. The advantage is that the QR-decomposition for M'' can be easily obtained from the QR-decomposition for M; since M'' is a $\operatorname{rank}(\ker M)$-update of M followed by row/column deletion updates, we can use the QR-update algorithm of [17, Section 6.5.1], with Givens rotation replaced by $\mathrm{GL}_2(\mathbb{Z}_p)$-elimination. The QR-update only requires $O(n^2)$ arithmetic operations when the kernel has rank 1. However, the QR-decomposition is only rank revealing given sufficient precision (see Example 2.2.10). We do not presently know how much precision is needed for this modification to work correctly.

4. The improved QR-iteration

Our proof of super-linear convergence in the QR-iteration depends on being able to convert the matrix to a size-sorted Hessenberg matrix. First, we give the standard Hessenberg algorithm for reference.

For attempting to compute a size-sorted Hessenberg matrix, we make two modifications to the standard Hessenberg algorithm. First, we start from the bottom and proceed upward rather than starting from the left and proceeding right. Secondly, we restrict the set of permutations in step 3 so that the sorted form is preserved. The reason we start from the bottom row in Procedure 4.2 is that *the procedure is guaranteed to produce a sorted Hessenberg matrix if $b = 1$*. This is because the condition in step 4 is vacuously false.

4.1. Super-linear separation.

In the case that $M := [A; \epsilon, B]$ is a size-sorted Hessenberg matrix, the separating entry ϵ will deflate superlinearly to 0 in the QR-iteration. We organize the proof of this statement into a sequence of three results.

Algorithm 4.1 `standard_hessenberg`

Input:
$M + O(p^N)$, an $n \times n$-matrix over $\mathbb{Z}_p$.
Output: A Hessenberg form H for M and a $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ such that $HU = UM + O(p^N)$

Set $U := I$
for $j = 1, \ldots, n-1$ **do**
  Find the minimal $i \in \{j+1, \ldots, n\}$ such that $|M\{i,j\}|$ is maximal
  Permute row $j+1$ and row i in M. Permute row $j+1$ and row i in U
  if $M\{j+1, j\} \neq 0$ **then**
    Set $U\{i,j\} := -M\{i,j\}/M\{j+1,j\}$ for $j+2 \leq i \leq n$
    Compute UM by using row $j+1$ to eliminate each $M\{i,j\}$ for $j+2 \leq i \leq n$
    Compute MU^{-1} by applying column operations
return M, U

Procedure 4.2 `attempt_sorted_hessenberg`

Input:
$M + O(p^N)$, an $n \times n$ size-sorted matrix over $\mathbb{Z}_p$.
The block sizes (a, b) for M.
Output: A sorted Hessenberg form H for M and a $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ such that $HU = UM + O(p^N)$

Set $U := I$
for $i = n, \ldots, 2$ **do**
  Find the maximal $j \in 1, \ldots, i-1$ such that $|M\{i,j\}|$ is maximal
  if $j \leq a$ and $a < i - 1$ **then**
    return `Fail`, M, U
  Permute column $i-1$ and column j in M. Permute column $i-1$ and column j in U
  if $M\{i, i-1\} \neq 0$ **then**
    Set $U\{i,j\} := M\{i,j\}/M\{i,i-1\}$ for $1 \leq j \leq i-2$
    Compute MU^{-1} by using column $i-1$ to eliminate each $M\{i,j\}$ for $1 \leq j \leq i-2$
    Compute UM by applying row operations
return `Success`, M, U

Lemma 4.1.1. *Let $M := [A; \epsilon, B]$ be a size-sorted Hessenberg matrix with block sizes $(n_A, *)$, and let $\mu \in \epsilon \cdot \mathbb{Z}_p$. Write $M - \mu I = Q_M R_M$ and $B - \mu I = Q_B R_B$. Then Q_M is block upper triangular modulo ϵ. Additionally, with*

$$M' := R_M Q_M + \mu I =: \left[\begin{array}{cc|c} & A' & * \\ \hline 0 & \epsilon' & \\ 0 & 0 & B' \end{array}\right], \quad \alpha := R_M\{n_A+1, n_A+1\},$$

we have that $|\epsilon'| = |\epsilon| \cdot |\alpha|$ and $|\alpha| \leq \max\{|\epsilon|, |R_B\{1,1\}|\}$.

Proof. Write $A - \mu I = Q_A R_A$. Then

$$(Q_A \oplus I)^{-1}(M - \mu I) = \left[\begin{array}{cc|c} & R_A & * \\ \hline 0 & \epsilon & \\ 0 & 0 & B - \mu I \end{array}\right].$$

Let $r = R_A\{n_A, n_A\} \in \mathbb{Z}_p$. Note that $|r| = |R_A\{n_A, n_A\}| \geq |\sigma_*(R_A)| = 1$ since μ is small and we have assumed that all of the small eigenvalues correspond to the block B. So $|r| = 1$.

The next operation in computing $M - \mu I = Q_M R_M$ is the elimination of the ϵ entry. The elementary row matrix for this step is $E := [I_A; -r^{-1}\epsilon, I_B]$, and the resulting intermediate matrix is

$$E \cdot (Q_A \oplus I)^{-1} \cdot (M - \mu I) =: [R_A; 0, C].$$

Writing $C = Q_C R_C$ and $M - \mu I = Q_M R_M$, we have that

$$Q_M = \left[\begin{array}{cc|c} \multicolumn{2}{c|}{Q_A} & * \\ \hline 0 & -r^{-1}\epsilon & \\ 0 & 0 & Q_C \end{array}\right], \quad R_M = \left[\begin{array}{cc|c} \multicolumn{2}{c|}{R_A} & * \\ \hline 0 & 0 & \\ 0 & 0 & R_C \end{array}\right].$$

As $R_C\{1,1\} = \alpha$ and $M' = R_M Q_M + \mu I = [A'; \epsilon', B']$, then by direct calculation $|\epsilon'| = |\alpha| \cdot |\epsilon|$. On the other hand, as $C \equiv B \mod \epsilon$ (since $\mu \equiv 0 \mod \epsilon$), we get that $|\alpha| \leq \max\{|\epsilon|, |R_B\{1,1\}|\}$, which concludes the proof. □

Corollary 4.1.2. *Let $M := [A; \epsilon, B]$ be a size-sorted Hessenberg matrix with block sizes $(*, m)$ such that $\|\chi_{\text{small}} - t^m\| \leq |\epsilon|$. Then after m QR-rounds, we obtain a matrix $M' := [A'; \epsilon', B']$ with $|\epsilon'| \leq |\epsilon|^2$.*

Proof. Let $\chi_M = \chi_{\text{big}}\chi_{\text{small}}$. By Lemma 3.0.4 we have $\|\chi_B - \chi_{\text{small}}\| \leq |\epsilon|$ and by assumption, $\|\chi_{\text{small}} - t^m\| \leq |\epsilon|$. Applying the Cayley-Hamilton theorem we then obtain:

$$-B^m \equiv \chi_B(B) - B^m \equiv \chi_{\text{small}}(B) - B^m \equiv 0 \pmod{\epsilon}.$$

Let $(Q_M^{(1)}, R_M^{(1)}), \ldots, (Q_M^{(m)}, R_M^{(m)})$ be the QR-pairs for the m QR-rounds, define $R_A^{(j)}, R_B^{(j)}$ by $[R_A; 0, R_B] := R_M^{(j)}$. Let $\delta^{(j)} := R_B^{(j)}\{1,1\}$ for each $1 \leq j \leq m$ and let

$$\mathcal{Q}_M^{(m)} := Q_M^{(1)} \ldots Q_M^{(m)}, \qquad \mathcal{R}_M^{(m)} := R_M^{(1)} \ldots R_M^{(m)}, \qquad \mathcal{R}_B^{(m)} := R_B^{(1)} \ldots R_B^{(m)}.$$

From Wilkinson's Lemma, $\mathcal{Q}^{(m)}\mathcal{R}^{(m)} = M^m$ and $M^m \equiv [A^m; 0, B^m] \pmod{\epsilon}$. As the R-factors are upper triangular, we have $\left|\prod_{j=1}^m \delta^{(j)}\right| = \left|\mathcal{R}_B^{(m)}\{1,1\}\right| \leq |\epsilon|$. By applying Lemma 4.1.1 to all of the QR-rounds, we have either $|\epsilon'| \leq |\epsilon|^2$ or

$$|\epsilon'| \leq \left(\left(\left(|\epsilon| \cdot \left|\delta^{(1)}\right|\right) \cdot \left|\delta^{(2)}\right|\right) \ldots\right) \cdot \left|\delta^{(m)}\right| \leq |\epsilon|^2. \qquad \square$$

In the proof of Corollary 4.1.2, we only needed that $\|B^m e_1\| \leq |\epsilon|$. Eran Assaf pointed out to us that we can compute $B^m e_1$ in $\mathcal{M}(m) \cdot \log_2 m$ operations, and efficiently forecast whether m QR-rounds will decrease the size of ϵ to $|\epsilon|^2$ – here $\mathcal{M}(m)$ denotes the number of operations needed to multiply two $m \times m$ matrices.

Corollary 4.1.3. *Let $M := [A; \epsilon, B]$ be a size-sorted Hessenberg matrix with block sizes $(*, m)$ and $|\epsilon| < 1$. Let $1 \leq \gamma \leq -\log_p \|\chi_{\text{small}} - t^m\|$ be a real value. Then after $(m\lceil \log_2 \log_p(\gamma) \rceil)$ QR-rounds, we obtain a size-sorted Hessenberg matrix $M' := [A'; \epsilon', B']$ with $|\epsilon'| \leq p^{-\gamma}$.*

Proof. Straightforward induction. □

4.2. **Trace shifting.** We show how to choose shifts μ such that $[A - \mu I; \epsilon, B - \mu I]$ satisfies the condition on the size of the small characteristic polynomial, or if it does not, we can prove that two clusters of small eigenvalues can be separated modulo ϵ.

Proposition 4.2.1. *Let $M := [A; \epsilon, B]$ be a sorted Hessenberg matrix with block sizes $(*, m)$ and let $\mu := \frac{1}{m}\text{trace}(B)$. Factor $\chi_M(t) = \chi_{\text{big}}\chi_{\text{small}}$ (with χ_A, χ_B equal to χ_{big}, $\chi_{\text{small}} \mod \epsilon$, respectively). If for all pairs of distinct roots λ_1, λ_2 of χ_{small}, we have $|\lambda_1 - \lambda_2| \leq |\epsilon|$, then $\|\chi_B(t - \mu) - t^m\| \leq |\epsilon|$. By contraposition, if $\|\chi_B(t - \mu) - t^m\| > |\epsilon|$, then there are some distinct roots λ_1, λ_2 of χ_{small}, such that $|\lambda_1 - \lambda_2| > |\epsilon|$.*

Proof. Let K be the field of definition of the eigenvalues of χ_M with ring of integers $\mathcal{O}_K$. Assume that for all pairs of distinct roots $\lambda_1, \lambda_2 \in O_K$ of χ_{small}, we have $|\lambda_1 - \lambda_2| \leq |\epsilon|$, *i.e.* $\lambda_1 \equiv \lambda_2 \mod \epsilon$. By Lemma 3.0.4, we have $\chi_B = \chi_{\text{small}} \mod \epsilon$, so $\mu = \frac{1}{m}\text{trace}(B) = \lambda_1 \mod \epsilon$. We compute that:

$$\begin{aligned} \chi_{\text{small}}(t - \mu) &\equiv \prod_i (t - \lambda_i - \mu), \\ &\equiv t^m \mod \epsilon. \end{aligned}$$

As $\chi_B = \chi_{\text{small}} \mod \epsilon$, we can conclude that $\|\chi_B(t - \mu) - t^m\| \leq |\epsilon|$. □

When $p \mid m$, there is a potential ambiguity in choosing the last digits of μ. However, since only finding the common leading digits of the eigenvalues is necessary, we may make some arbitrary choice and convergence will be unaffected beyond the possibility of accidentally choosing a better shift than expected. *In the specific (very common) case that $m = 1$, we will always choose a good shift and the precision of ϵ will at least double at every step.* To clarify what we mean by common, see Remark 1.2.4.

Based on various experiments, the condition that $\|\chi_{\text{small}} - t^m\| \leq |\epsilon|$ is genuinely necessary to ensure quadratic convergence. We remark that the converse of Proposition 4.2.1 is false; consider

$$M := \left[1; p^2, \begin{bmatrix} p & 0 \\ 0 & -p \end{bmatrix} + O(p^2)\right] \quad p \neq 2.$$

We have with $\epsilon := p^2$ that $\chi_B(t) \equiv (t-p)(t+p) \equiv t^2 + O(p^2)$, but $(-p) \not\equiv p \pmod{p}^2$.

Proposition 4.2.2. *Let $M := [A; \epsilon, B]$ be a size-sorted Hessenberg matrix, let $m = n_B$ and let $\lambda_1, \ldots, \lambda_m$ be the small eigenvalues of M. Let $\mu := \frac{1}{m}\operatorname{trace}(B)$. If $\eta := \max_{i,j} |\lambda_i - \lambda_j| \leq |\epsilon|$, then after m QR-rounds with shift μ we obtain a size-sorted Hessenberg matrix $[A_{\text{next}}; \epsilon_{\text{next}}, B_{\text{next}}]$ such that $|\epsilon_{\text{next}}| \leq |\epsilon^2|$. After at most $(m\lceil \log_2(-\log_p \eta)\rceil)$ rounds, the obtained $[A_{\text{next}}; \epsilon_{\text{next}}, B_{\text{next}}]$ is such that $|\epsilon_{\text{next}}| \leq \eta$.*

Proof. The result follows from Proposition 4.2.1, Corollary 4.1.2, and Corollary 4.1.3. □

4.3. **Further properties of the QR-iteration.** In this subsection, we prove some further results about the QR-iteration. This section is not necessary to implement our main algorithm, but is intended to explain some patterns we have observed in computing several examples. Some heuristics are supported by these results.

Separating eigenvalues would be useful to continue converging quickly. The only way we presently are aware of doing this is to compute some approximation of the characteristic polynomial. We have already seen that low precision approximations, such as $\chi_M \pmod{p}$, provide a mean to separate the eigenvalues. We explain how to efficiently approximate some factor of the characteristic polynomial during a QR-iteration. Unfortunately, it is possible that this approximation is not sufficient to separate the roots. If a separation of the roots is detected, then we can continue running the QR-iteration using the refined shifts.

We denote by $P_M(m)$ the matrix $[e_1\ Me_1\ \ldots\ M^{m-1}e_1]$. If $B + O(p^N) \in \mathrm{M}_n(\mathbb{Z}_p)$ is topologically nilpotent, the matrix $P_B(m)$ is often not given at a flat absolute precision; the i-th column is actually known at absolute precision $N - \log_p \left\|B^i e_1\right\|$. By Wilkinson's lemma, columns of the matrix $P_B(m)$ can be cached during a QR-iteration, so the cost of constructing the matrix is negligible.

Lemma 4.3.1. *Let $M \in \mathrm{M}_n(\mathbb{Z}_p)$ be a Hessenberg matrix. Then for all $m \geq 1$, we have that $P_M(m)$ is upper triangular, and for each $1 \leq i \leq n$, we have $|P_M(m)\{i,i\}| \geq |P_M(m)\{i',j'\}|$ for all $i', j' \geq i$.*

Proof. Triangularity is obvious. Let $\alpha := P_M(m)\{i,i\}$ be a diagonal entry. If $|\alpha| = 1$ there is nothing to do, and if $|\alpha| < 1$ we have that $M^i e_1$ is a $\mathbb{Z}_p$-linear span of $e_1, Me_1, \ldots, M^{i-1}e_1$ modulo α. When $j' > n$, we have $M^{j'}e_1$ is a span of the columns of $P_M(n)$ by the Cayley-Hamilton Theorem. □

Corollary 4.3.2. *The matrix $P_M(m)$ admits a factorization $P_M(m) = DQ_M(m)$, where D is a diagonal matrix such that $|D\{i,i\}| \geq |D\{i+1,i+1\}|$ and where $Q_M(m) \in \mathrm{GL}_n(\mathbb{Z}_p)$.*

Using Corollary 4.3.2, we can determine an approximation to a factor of χ_{small} provided that either some $D\{i,i\}$ is very small (in which case, the orbit of e_1 is nearly a proper invariant subspace), or provided that no $D\{i,i\}$ is too small (meaning the matrix $P_A(m)$ is reasonably well-conditioned). We believe that a more precise statement of what we can determine from this approximation to χ_{small} is an interesting problem for future study.

4.4. **The QR-algorithm.** We give the fast version of the QR-algorithm, given as Algorithm 4.3. The conditional statement on Line 11 should be interpreted as "while the iteration is still converging super-linearly".

Proposition 4.4.1. *Let $M := [A; \epsilon, B]$ be a size-sorted Hessenberg matrix, let $m = n_B$ and let $\lambda_1, \ldots, \lambda_m$ be the small eigenvalues of M. If $\eta := \max_{i,j} |\lambda_i - \lambda_j| \leq |\epsilon|$, then after m QR-rounds we obtain a size-sorted Hessenberg matrix $[A_{\text{next}}; \epsilon_{\text{next}}, B_{\text{next}}]$ such that $|\epsilon_{\text{next}}| \leq |\epsilon^2|$. Each round*

Algorithm 4.3 `QR_Iteration` (Fast version)

Input:
$H + O(p^N)$, an $n \times n$-matrix over $\mathbb{Z}_p$ in size-sorted Hessenberg form.
$\chi_{H,p}$, the characteristic polynomial of $M \pmod p$.

Output: A block triangular form T for H, and matrix V so that $HV = VT + O(p^N)$.

1: Set m to be the multiplicity of 0 in $\chi_{H,p}$.
2: Set $[A; \epsilon, B] := H$
3: Set $\epsilon_{\text{old}} := 1$
4: **while** `true` **do**
5: **for** $j = 1, \ldots, m$ **do**
6: Set $\mu := m^{-1} \operatorname{trace}(B)$. Adjust precision if needed.
7: Factor $QR := H - \mu I$
8: Set $H := RQ + \mu I$
9: Set $[A; \epsilon, B] := H$
10: Set $V := Q^{-1}V$
11: **if** $|\epsilon| > |\epsilon_{\text{old}}|^2$ **then**
12: **return** `Fail`, H, V
13: **else if** $\epsilon = 0 \pmod{p^N}$ **then**
14: **return** `Success`, H, V
15: **else**
16: Set $\epsilon_{\text{old}} := \epsilon$.

Line(s)	Cost per line (leading term)	
4	$\lceil \log_2 N \rceil$ iterations	
– 5	m iterations	
– – 7,8, & 10	$\frac{1}{2}n^2 + \frac{1}{2}n^2 + n^2$	In parallel
Total (main term):	$2n^2 m \lceil \log_2 N \rceil$	

TABLE 1. Table of costs for the QR-algorithm.

uses $2n^2 + o(n^2)$ *operations of* $\mathbb{Q}_p$ *arithmetic. After at most* $(m\lceil \log_2(-\log_p \eta)\rceil)$ *rounds, the obtained* $[A_{\text{next}}; \epsilon_{\text{next}}, B_{\text{next}}]$ *is such that* $|\epsilon_{\text{next}}| < \eta$.

Proof. The result is obtained by combining Proposition 4.2.2 and tabulating the costs in Table 1. □

5. The main algorithm

In this section, we describe the main algorithm (Algorithm 5.1) and prove the main theorem. Though our main theorem is concerned with matrices whose eigenvalues are all defined in $\mathbb{Q}_p$, we introduce some terminology to state more precisely how our algorithm performs in general

Definition 5.0.1. We say that a matrix is in *weak block Schur form* if it is block upper triangular and for each block B, either the characteristic polynomial of B has no roots in $\mathbb{Q}_p$ or there is a $\lambda \in \mathbb{Q}_p$ such that $B - \lambda I$ is topologically nilpotent.

Note that the weak block Schur form can be converted to a block Schur form by applying the eigenvector methods [4, 8] to the diagonal blocks, and then applying the resulting change of basis to the whole matrix. If the characteristic polynomial of M modulo p is square-free and splits completely, the weak block Schur form is a Schur form.

Note that each of the matrix multiplication steps in Algorithm 5.1 can be combined into the preceding step, so do not actually contribute to the complexity; we separated out the update steps for clarity.

Algorithm 5.1 Main algorithm

Input:
$M + O(p^N)$, an $n \times n$-matrix over $\mathbb{Q}_p$.

Output: A weak block Schur form T for M, and matrix U such that $MU = UT + O(p^N)$.

1: Set $d := \|M\|$, $M := d^{-1} \cdot M$
2: Set $M, U :=$ `sorted_form(M)`
3: Set S to be the block sizes of M
4: Set `retcode`, $M, U_1 :=$ `attempt_sorted_hessenberg`(M, S)
5: Update $U := UU_1$
6: Compute $\chi_{M,p}$
7: **while** M has an eigenvalue defined over $\mathbb{Z}_p$ **do**
8: Choose $\mu \in \mathbb{Z}_p$ such that $\mu \pmod p$ is a root of the characteristic polynomial of the bottom-right block of $M \pmod p$
9: Apply one QR-round to M with shift μ
10: **if** $M - \mu I \pmod p$ is a size-sorted Hessenberg matrix **then**
11: Set `retcode2`, $M, U_2 :=$ `QR_Iteration`$(M - \mu I, \chi_{M,p})$
12: Set $M := M + \mu I$
13: Update $U := UU_2$
14: **else**
15: Set `retcode` := `Fail`
16: **if** `retcode` == `Fail` **or** `retcode2` == `Fail` **then**
17: Apply a fallback method (we use Algorithm 2.1, and obtain the output M, U_3)
18: Update $U := UU_3$
19: **return** M, U
20: Deflate M to be the top-left block (thereby reducing the size of M)
21: Reset M to be the full-sized matrix
22: **return** $d \cdot M, U$.

5.1. **Proof of the Main Theorem.** We now prove our main theorem on the behaviour of Algorithm 5.1 in the special case of a matrix with n eigenvalues in $\mathbb{Z}_p$ that are simple modulo p.

Theorem 5.1.1. *Let $M \in \mathrm{M}_n(\mathbb{Z}_p)$ be a matrix whose entries are known with error $O(p^N)$. If the characteristic polynomial of M modulo p is square-free and factors completely then Algorithm 5.1 computes a Schur form T and a matrix $U \in \mathrm{GL}_n(\mathbb{Z}_p)$ such that $MU = UT + O(p^N)$ in at most $\frac{2}{3}n^3 \log_2 N + o(n^3 \log_2 N)$ arithmetic operations in $\mathbb{Z}_p$ at N-digits of precision. In particular, T reveals all the eigenvalues of M with error $O(p^N)$. An additional $O(n^3)$ arithmetic operations in $\mathbb{Q}_p$ is then enough to compute a $\mathbb{Q}_p$-basis of eigenvectors with coefficients in $\mathbb{Z}_p$.*

Proof. After step 2, we may assume that our matrix is of the form

$$M \equiv \begin{bmatrix} A & * & \cdots & * \\ & B_1 & & \vdots \\ & & \ddots & * \\ & & & B_r \end{bmatrix} \pmod p,$$

where $\chi_A \pmod p$ has no linear factors, and every $\chi_{B_j}(t) \equiv (t - \lambda_j)^{m_j} \pmod p$ for some $\lambda \in \mathbb{F}_p$. By our assumption on χ_M, we see that the A block is empty and B_r is a block of size 1 in $M \pmod p$. We see that step 4 will produce a sorted Hessenberg matrix of the form $[A; \epsilon, b_r]$ and that the condition in step 10 is satisfied. By Proposition 4.4.1, step 11 will transform M to a matrix of the form $[A'; 0, \lambda_r]$. Additionally, step 11 will preserve the Hessenberg form.

We now look at the deflated instance where $M'' = A'$. Specifically, we will show that the condition in step 10 is satisfied. Write

$$M'' \equiv \begin{bmatrix} B''_1 & \cdots & * \\ & \ddots & \vdots \\ & & B''_{r''} \end{bmatrix} \pmod{p},$$

where by definition the subdiagonal entries of each B''_j are non-zero modulo p. By the assumption on χ_M, we have that $\chi_{B''_{r''}} \pmod{p}$ has a simple root $\bar{\mu}$ over $\mathbb{F}_p$. We choose a lift $\mu \in \mathbb{Z}_p$ for $\bar{\mu}$.

For a Hessenberg matrix H, we have with $H = QR$ a QR-decomposition that $|R\{i,i\}| \geq |H\{i+1,i\}|$. Thus, after one QR-round with shift μ we have that the bottom row of M'' is congruent to 0 modulo p. Since $\bar{\mu}$ is a simple root of the characteristic polynomial, we additionally have that M is in sorted Hessenberg form. Thus, step 10 succeeds to produce a size-sorted Hessenberg matrix. We now see that the algorithm produces a Schur form for M by induction.

By Proposition 4.4.1, we see that each execution of step 11 consists of $\log_2(N)$ QR-rounds, after which the subdiagonal ϵ converges to $0 + O(p^N)$. The total cost for this is $2n^2 \log(N) + o(n^2 \log(N))$. Since deflation reduces the number of rows/columns of the input matrix by 1, we see repeated applications of step 11 require a total of $\frac{2}{3}n^3 \log(N) + o(n^3 \log(N))$ arithmetic operations in $\mathbb{Z}_p$. Finally, to compute the eigenvectors, only n triangular systems are to be solved, for a total of $O(n^3)$ arithmetic operations in $\mathbb{Q}_p$ (there may be some divisions by powers of p). □

6. Practicality and Implementation

In this section, we give some timings for our `Julia` implementation, available at:

`https://github.com/a-kulkarn/Dory`

Our benchmarking results are listed in Tables 3 and 4. We also include the old timings from [8] for the sake of reference (Table 2), however, the updates to the dependencies and the change in hardware means the comparison is not pure. Timings are based on random matrices, where each entry is a randomly sampled p-adic number in `PadicField(`p`,`N`)` (more precisely, a uniformly random integer in $[0, p^N - 1]$).

Matrix size (n)	Time (s) (power iteration)	Time(s) (block schur form)	Time (s) (classical)
10	0.0029	0.010	0.0008
100	0.9774	3.390	3.2600
200	6.7920	24.2771	51.2573
300	36.0114	166.4447	258.0104

Table 2. Timings from [8]. ($\mathbb{Q}_p :=$ `PadicField(7,10)`)

Matrix size (n)	Time (s) (power iteration)	Time(s) (block schur form)	Time (s) (classical)
10	0.0017	0.0524	0.0006
100	0.5386	1.4558	2.0400
200	3.7068	10.1043	31.1456
300	20.4178	52.0343	158.4332

Table 3. Timings with improved QR. ($\mathbb{Q}_p :=$ `PadicField(7,10)`, simple roots over $\mathbb{F}_p$)

Timings were conducted by using the `time()` function. An average of 10 samples were used per comparison, with each method receiving the same inputs. We omit from the timings an extra execution of each function at the beginning which triggers `Julia`'s compiler. The code to execute the comparisons is found in `Dory/test/timings.jl` and `Dory/test/timings2.jl`.

Matrix size (n)	Time (s) (power iteration)	Time(s) (block schur form)	Time (s) (classical)
10	0.0125	0.0196	0.0060
100	6.9100	14.9795	19.7082
200	44.5217	39.6243	337.6393

TABLE 4. Timings with improved QR, more precision. ($\mathbb{Q}_p :=$ `PadicField(41,100)`)

ACKNOWLEDGEMENTS

The authors would like to thank the mathematics department at TU Kaiserslautern for sponsoring the visit of the second author. We would also like to thank Eran Assaf and John Voight for their especially insightful comments.

REFERENCES

[1] John D. Dixon, *Exact solution of linear equations using p-adic expansions*, Numer. Math. **40** (1982), no. 1, 137–141, DOI 10.1007/BF01459082. MR681819

[2] P. Panayi, *Computation of Leopoldt's p-adic regulator*, PhD thesis, University of East Anglia, 1995.

[3] Xavier Caruso, David Roe, and Tristan Vaccon, *p-adic stability in linear algebra*, ISSAC'15—Proceedings of the 2015 ACM International Symposium on Symbolic and Algebraic Computation, ACM, New York, 2015, pp. 101–108. MR3388288

[4] ———, *Characteristic polynomials of p-adic matrices*, ISSAC'17—Proceedings of the 2017 ACM International Symposium on Symbolic and Algebraic Computation, ACM, New York, 2017, pp. 389–396. MR3703711

[5] Kiran S. Kedlaya, *p-adic differential equations*, Cambridge Studies in Advanced Mathematics, vol. 125, Cambridge University Press, Cambridge, 2010. MR2663480

[6] ———, *Counting points on hyperelliptic curves using Monsky-Washnitzer cohomology*, J. Ramanujan Math. Soc. **16** (2001), no. 4, 323–338. MR1877805

[7] Jérémy Berthomieu and Romain Lebreton, *Relaxed p-adic Hensel lifting for algebraic systems*, ISSAC 2012—Proceedings of the 37th International Symposium on Symbolic and Algebraic Computation, ACM, New York, 2012, pp. 59–66, DOI 10.1145/2442829.2442842. MR3206287

[8] Avinash Kulkarni, *Solving p-adic polynomial systems via iterative eigenvector algorithms*, Linear and Multilinear Algebra **0** (2020), no. 0, 1-22, DOI 10.1080/03081087.2020.1743633.

[9] W. H. Schikhof, *Ultrametric calculus*, Cambridge Studies in Advanced Mathematics, vol. 4, Cambridge University Press, Cambridge, 2006. An introduction to p-adic analysis; Reprint of the 1984 original [MR0791759]. MR2444734

[10] Jason Fulman, *Random matrix theory over finite fields*, Bull. Amer. Math. Soc. (N.S.) **39** (2002), no. 1, 51–85, DOI 10.1090/S0273-0979-01-00920-X. MR1864086

[11] Xavier Caruso, *Computations with p-adic numbers*, Vol. 5, CIRM, 2017 (en).

[12] Oscar Zariski and Pierre Samuel, *Commutative algebra. Vol. II*, Springer-Verlag, New York-Heidelberg, 1975. Reprint of the 1960 edition; Graduate Texts in Mathematics, Vol. 29. MR0389876

[13] J. H. Wilkinson, *Convergence of the* LR*,* QR*, and related algorithms*, Comput. J. **8** (1965), 77–84, DOI 10.1093/comjnl/8.3.273. MR183108

[14] Xavier Caruso, David Roe, and Tristan Vaccon, *Division and Slope Factorization of p-Adic Polynomials*, ISSAC'16—Proceedings of the 2016 ACM International Symposium on Symbolic and Algebraic Computation, ACM, New York, 2016, pp. 159–166.

[15] Jordi Guàrdia, Enric Nart, and Jesus Montes, *The Montes project,* `http://montesproject.blogspot.com/`.

[16] Jordi Guàrdia, Enric Nart, and Sebastian Pauli, *Single-factor lifting and factorization of polynomials over local fields*, J. Symbolic Comput. **47** (2012), no. 11, 1318–1346, DOI 10.1016/j.jsc.2012.03.001. MR2927133

[17] Gene H. Golub and Charles F. Van Loan, *Matrix computations*, 4th ed., Johns Hopkins Studies in the Mathematical Sciences, Johns Hopkins University Press, Baltimore, MD, 2013. MR3024913

[18] Xavier Caruso, David Roe, and Tristan Vaccon, *Tracking p-adic precision*, LMS Journal of Computation and Mathematics **17** (2014), no. A, 274–294.

DARTMOUTH COLLEGE, HANOVER, NH 03755, USA
E-mail address: `avinash.a.kulkarni@dartmouth.edu`

UNIV. LIMOGES, CNRS, XLIM, UMR 7252, F-87000 LIMOGES, FRANCE
E-mail address: `tristan.vaccon@unilim.fr`