POINCARÉ NORMAL FORM OF RIEMANN MATRICES IN SAGEMATH

CALEB ARYEE

Contents

0. Prelude

Personal information:

- *Name*: Caleb Aryee
- *Email*: ca2004mail@gmail.com
- *Location*: London/Edinburgh
- *University*: University of Edinburgh, Mathematics

Background:

- *Technical skills, education and experience*: I am currently a third-year undergraduate student in mathematics at the University of Edinburgh. My academic work involves a strong emphasis on formal mathematical reasoning, especially in real and complex analysis, algebra, and topology. I am highly comfortable with rigorous mathematical proofs.

  From a computational standpoint, I have experience with Python and have worked with SageMath for various coursework and independent mathematical investigations. For example, in the

course Honours Algebra, I was tasked with investigating the spectrum of a 'nice' class of matrices. In specific, I defined Sage functions to construct these matrices in `QQ` and from observation of the resulting eigenvalues and eigenvectors in both `QQ` and `SR`, I formulated a conjecture on the spectrum of these matrices. I then proved this conjecture with the aid of Sage. Similarly, in the same course, I investigated the centralisers of another 'nice' class of matrices. In specific, I investigated the dimensions of the centralisers of these matrices and their dimensions in relation to the Jordan decomposition of these matrices. Using Sage, I again formulated a conjecture on the dimensions of the centralisers. In this course, relating to coursework in SageMath, I have (to this date) scored an average of 96%.

My programming ability is intermediate to advanced, particularly as it relates to scientific computation and symbolic mathematics. I am particularly interested in implementing algorithms with theoretical significance, such as those in the context of Riemann surfaces and period matrix reduction.

- *Suitability for project*: As a mathematics undergraduate with a deep appreciation for formalism and clarity in both proof-writing and programming, I bring a rigorous approach to mathematical computing. My familiarity with SageMath, together with my ability to understand and implement advanced mathematical algorithms, makes me well-suited for contributing to projects focused on symbolic computation, Riemann surfaces, or computational algebraic geometry.
- *Operating System*: I currently use Windows. While this is my primary environment, I plan to use WSL (Windows Subsystem for Linux) to support SageMath and related tools effectively.
- *Open source contributions*: This will be my first time contributing to an open-source project. However, in writing the proposal, I have gained a deeper understanding of the SageMath codebase and its structure. From this, I have wrote an issue (#39903) on the SageMath GitHub repository to discuss a minor issue in documentation of the `differential_basis_baker()` function.
- *Past projects*: My programming experience is primarily academic and coursework-based. While I do not maintain personal repositories outside university work, I have applied programming in mathematical modelling, algorithm design, and exploratory computation in support of my studies.
- *SageMath usage*: I have been a SageMath user since being introduced to it during my undergraduate studies. I use it primarily for experimentation with algebraic structures.

Project information:

- *Project title*: Poincaré Normal Form of Riemann Matrices
- *Project length*: Long (350 hours)
- *Project relationship*: My strong background in mathematics and SageMath, makes this project a compelling application of my skills and aligns with my academic goals. It is an extension of the work done in [2] and will be a valuable addition to the SageMath library. I am particularly interested in the intersection of algebraic geometry and computational methods and this project provides an opportunity to explore this area in depth. I am excited about the potential impact of this work on the SageMath community and the broader mathematical community. I am also interested in the practical applications of Riemann surfaces and theta functions in number theory and algebraic geometry, and this project will allow me to explore these connections further.

1. Introduction

A Riemann surface is a one-dimensional complex manifold which provides a setting for the study of complex functions. Associated to the surface are a variety of algebraic and analytic invariants among which the Riemann matrix and associated theta functions are of particular interest.

1.1. **Riemann surfaces.** We consider a compact Riemann surface X of genus g and its associated period matrix $\begin{pmatrix} E & Z \end{pmatrix}$ where Z satisfies the Riemann relations:

$$Z^{\mathrm{T}} = Z, \quad \text{and} \quad \Im(Z) > 0.$$

We call Z the Riemann matrix of X and the matrix E is the identity matrix of size g. Indeed, the period matrix of any X may take this form after choosing the canonical homology basis $\{a_1, \dots, a_g, b_1, \dots, b_g\}$ of $H_1(X, \mathbb{Z})$ and choosing a basis of holomorphic 1-forms of $\Omega_1(X)$ such that for $1 \le i, j \le g$,

$$\int_{a_i} \omega_j = \delta_{ij},$$

where δ_{ij} is the Kronecker delta.

From this matrix Z, we can define the Jacobian variety of X, a complex torus defined by

$$\mathrm{Jac}(X) = \mathbb{C}^g/(\mathbb{Z}^g + Z\mathbb{Z}^g).$$

We call $\mathbb{Z}^g + Z\mathbb{Z}^g$ the period lattice.

1.2. **Theta functions.** Let Z be the Riemann matrix of a compact Riemann surface X of genus g. The theta function $\theta(\mathbf{z}, Z)$ is defined as the following series:

$$\theta(\mathbf{z}, Z) = \sum_{\mathbf{n} \in \mathbb{Z}^g} \exp\left(\pi i \mathbf{n}^{\mathrm{T}} Z \mathbf{n} + 2\pi i \mathbf{n}^{\mathrm{T}} \mathbf{z}\right) \tag{1.2.1}$$

where $\mathbb{z} \in \mathbb{C}^g$.

1.3. **Poincaré reduction.** If $\mathrm{Jac}(X)$ is decomposable, it is isogenous (or isomorphic) to a product of Jacobians of lower dimension. It follows that if by isogeny, $\mathrm{Jac}(X) \sim J_1 \times J_2$ for example, the Riemann matrix Z is reducible. This means that there exists a symplectic transformation such that under this transformation, Z is represented by the matrix

$$Z' = \begin{pmatrix} Z_1 & Q \\ Q^{\mathrm{T}} & Z_2 \end{pmatrix}$$

where Z_1 and Z_2 are lower dimensional Riemann matrices and Q is a matrix of rationals. When this occurs, the theta function can be expressed as a product of lower-dimensional theta functions. For example, if $\mathrm{Jac}(X) \cong J_1 \times J_2$ where J_1 and J_2 are abelian varieties of dimension g_1 and g_2 (with $g = g_1 + g_2$) respectively, then there is a change of basis via a symplectic transformation T such that the Riemann matrix Z of X is represented as

$$Z' = \begin{pmatrix} Z_1 & 0 \\ 0 & Z_2 \end{pmatrix}$$

where Z_1 and Z_2 are the Riemann matrices of J_1 and J_2 respectively. If $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2)^{\mathrm{T}}$, then the theta function can be factorised as follows:

$$\begin{aligned}
\theta(\mathbf{z}, \Omega') &= \sum_{(\mathbf{n}_1, \mathbf{n}_2) \in \mathbb{Z}^{g_1} \times \mathbb{Z}^{g_2}} \exp\left(\pi i \begin{pmatrix} \mathbf{n}_1 \\ \mathbf{n}_2 \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} \Omega_1 & 0 \\ 0 & \Omega_2 \end{pmatrix} \begin{pmatrix} \mathbf{n}_1 \\ \mathbf{n}_2 \end{pmatrix} + 2\pi i \begin{pmatrix} \mathbf{n}_1 \\ \mathbf{n}_2 \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \end{pmatrix}\right) \\
&= \sum_{\mathbf{n}_1 \in \mathbb{Z}^{g_1}} \exp\left(\pi i \mathbf{n}_1^{\mathrm{T}} \Omega_1 \mathbf{n}_1 + 2\pi i \mathbf{n}_1^{\mathrm{T}} \mathbf{z}_1\right) \sum_{\mathbf{n}_2 \in \mathbb{Z}^{g_2}} \exp\left(\pi i \mathbf{n}_2^{\mathrm{T}} \Omega_2 \mathbf{n}_2 + 2\pi i \mathbf{n}_2^{\mathrm{T}} \mathbf{z}_2\right) \\
&= \theta(\mathbf{z}_1, \Omega_1)\theta(\mathbf{z}_2, \Omega_2).
\end{aligned}$$

We say a period matrix $\begin{pmatrix} E & Z \end{pmatrix}$ admits reduction if it satisfies

$$H \times \begin{pmatrix} E & Z \end{pmatrix} = \Pi \times M \tag{1.3.1}$$

where Π is an $m \times 2m$ matrix of complex numbers, H is an $m \times n$ matrix of complex numbers of maximal rank, and M is a maximal rank $2m \times 2n$ matrix of complex numbers also of maximal rank.

In general, by theorem of Weierstrass and Poincaré, if $\begin{pmatrix} E & Z \end{pmatrix}$ is an $n \times 2n$ period matrix of a compact Riemann surface which admits reduction, then there exists an $n \times n$ invertible matrix A of complex numbers and a $2n \times 2n$ symplectic unimodular matrix T such that

$$\begin{pmatrix} E & Z \end{pmatrix} \times T = A \times \begin{pmatrix} E_1 & 0 & Z_1 & Q \\ 0 & E_2 & Q^{\mathrm{T}} & Z_2 \end{pmatrix} \tag{1.3.2}$$

where E_1 and E_2 are the identity matrices of appropriate size and Z_1 and Z_2 are the Riemann matrices of dimension $m \times m$ and $(n-m) \times (n-m)$ respectively.

The matrix involving Z_1 and Z_2 arises from a normal form for M. That is, for $1 \leq m < n$, and for a $2m \times 2n$ matrix of maximal rank M such that $MJ^{\mathrm{T}}M$ where

$$J = \begin{pmatrix} 0 & E \\ -E & 0 \end{pmatrix},$$

then, $M = SNT$ where S is a $2m \times 2m$ invertible matrix of integers, T is a $2n \times 2n$ symplectic unimodular matrix, and N is given by

$$N = \begin{pmatrix} E_1 & 0 & 0 & 0 \\ 0 & X & \Delta & 0 \end{pmatrix}$$

where E_1 is the identity matrix of size m, Δ is a diagonal matrix of integers, and X is an $m \times (n-m)$ matrix where $(X)_{jj} = 1$ for all $j \leq r$ for $0 \leq r \leq n-m$, and the remaining entries are zero.

Thus, in order to factorise the theta function, we must compute the normal form of M and the process of this is known as Poincaré reduction.

1.4. **Project synopsis.** The goal of this project is to implement a way to compute the Poincaré normal form of M by determining the matrix T from M in SageMath which allows for the decomposition of the associated theta function.. Specifically, we implement the methods described by Martens in [2]. We aim to equip the SageMath computational framework with practical tools for detecting and performing such reductions.

Practically, a factorised theta function can be evaluated much faster than a single high-dimensional theta. Theta function computation is notoriously expensive as genus grows, since the summation involves $\mathbb{Z}^g$ lattice points. If $\theta(\mathbf{z}, Z) = \theta_1(\mathbf{z}_1, Z_1)\theta_2(\mathbf{z}_2, Z_2)$ for a decomposition $Z \sim \mathrm{diag}(Z_1, Z_2)$, one can compute the two lower-dimensional theta functions separately. In many cases, this reduces complexity from exponential in g to (roughly) the sum of exponentials in g_1 and g_2. For instance, a genus-4 theta might factor into two genus-2 thetas, potentially cutting the cost dramatically. This project thus has performance implications in SageMath's ability to handle higher-genus curves by breaking hard computations into smaller pieces.

Ultimately, this implementation enables the decomposition of the associated theta function. It is important to note that while the reduction of the Riemann matrix allows us to explicitly obtain the Riemann matrix of the factors and compute their invariants to identify them, it is not strictly necessary for identifying decomposable Jacobians. Instead, one can simply check the reduction condition provided by Martens. Nonetheless, this implementation will provide a valuable tool for those wishing to perform the reduction and explicitly work with the factor's Riemann matrices.

Importantly, implementing Poincaré reduction fills a current gap in SageMath's algebraic geometry and number theory toolkit. SageMath can already compute Riemann matrices numerically and evaluate theta functions via Nils Bruin's package (discussed below). Adding reduction capabilities now means Sage can not only compute these objects but analyse their structure. This elevates SageMath from a purely computational tool to a discovery tool in this area of mathematics.

2. Existing Functionality in SageMath

SageMath offers substantial capabilities for working with Riemann surfaces. This project aims to build upon this foundation, ensuring SageMath remains at the forefront of research applications in this area.

The existing codebase provides essential tools for defining Riemann surfaces, computing period matrices, and performing related calculations. We will leverage some of these tools, while others, though valuable, lie outside the immediate scope of this project.

Specifically, the implementation of Riemann surfaces is within `sage/schemes/riemann_surfaces/riemann_surface.py`, where we have the class `RiemannSurface` which provides the core functionality for working with Riemann surfaces defined by bivariate polynomials. Additionally, we compute the period matrix using `riemann_matrix()`, normalise period matrices and also compute bases for holomorphic 1-forms $\Omega^1(X)$, homology bases for $H_1(X, \mathbb{Z})$ and we may also integrate the given 1-forms. SageMath also offers functionality for computing the Abel-Jacobi map and the endomorphism ring of the Jacobian.

3. Missing Functionality in SageMath

3.1. **Reduction via the Poincaré reducibility theorem.** If $\mathrm{Jac}(X)$ is decomposable, then $\begin{pmatrix} E & Z \end{pmatrix}$ admits reduction.

Reduction check of $\mathrm{Jac}(X)$. There is currently no functionality in SageMath to check if this is the case. If $\mathrm{End}(\mathrm{Jac}(X))$ extended over $\mathbb{Q}$ contains non-trivial idempotents, that is there exists a non-trivial idempotent e such that $e^2 = e$, then $\mathrm{Jac}(X)$ is decomposable. As the endomorphism ring is computed already, checking the condition $e^2 = e$ is quick and exact.

From the endomorphism basis of $\mathrm{End}(\mathrm{Jac}(X))$, which returns a list of matrices $\{R_1, \ldots, R_k\}$, we check for the trivial case where $k = 1$, with $R_1 = I_g$ where the Jacobian is irreducible. Otherwise. we check for the existence of non-trivial idempotents in $A = \mathrm{End}(\mathrm{Jac}(X)) \otimes \mathbb{Q}$.

Computing M. For each pair of matrices R_i and R_j, we compute the matrix $P_{ij} = R_i R_j$ in order to determine how the basis elements interact. As $P_{ij} \in A$, it is a unique linear combination of the basis elements. We then find the values λ_i such that $P_{ij} = \sum \lambda_i R_i$. Now, for a general element $R \in A$, we can write $R = \sum \mu_i R_i$ for some $\mu_i \in \mathbb{Q}$ and compute the matrix $R^2 - R$ in terms of μ_i. For each basis element, we obtain k polynomial equations which may be solved by creating the ideal I in Y generated by the polynomials and then computing the Gröbner basis G of I. If $G = \{1\}$, then the only solution is the trivial one and the Jacobian is irreducible. However, if there are non-trivial solutions, we construct the corresponding matrix R using the solutions μ_i and double check that $R^2 = R$.

Selecting a non-trivial idempotent R, we compute the rank m of R and check if $m/2$ is an integer and that $1 \leq m < g$. If this is the case, we rescale R so that the entries of R are integer and compute the integer basis for the column space of R using the Hermite Normal Form of its transpose. The non-zero rows form M. Essentially, M is the integer matrix which acts on the standard basis of the homology lattice $\mathbb{Z}^{2g}$ and selects a rank $2m$ sublattice which corresponds precisely to the homology group $H_1(J_1, \mathbb{Z})$ of the subvariety J_1 into which the Jacobian J decomposes. The rows of M form an integer basis for the homology sublattice $H_1(J_1, \mathbb{Z})$ viewed in $H_1(\mathrm{Jac}(X), \mathbb{Z})$.

3.2. **Computing the Poincaré normal form of** M**.** In accordance to the process shown in Martens, we have an exact algorithm to compute the Poincaré normal form of M. We use the symplectic operations specified in section 2 of [2] and GCD arguments (Euclidean algorithm via matrix operations), to make the top m rows of M become $\begin{pmatrix} E_1 & 0 & 0 & 0 \end{pmatrix}$. Similarly, with use of the Picard trick, other operations and cleaning up, we make the bottom m rows of M become $\begin{pmatrix} 0 & X & \Delta & 0 \end{pmatrix}$.

From this, we may reduce the period matrix as needed.

3.3. **Evaluation of theta functions.** Currently, there is no extensive functionality of theta function evaluation in SageMath. However, we may integrate the code from Nils Bruin's `RiemannTheta` package. We note that SageMath is currently the only open-source computer algebra system that computes Riemann matrices. The existing implementation within SageMath is highly capable, providing fast and arbitrary precision computation of Riemann matrices. While other implementations of theta functions exist, they

rely on Riemann matrices computed by SageMath. This project not only offers theoretical value by enabling the study of decomposable Jacobians but also strengthens SageMath's position as a crucial tool for theta function computations.

4. Implementation Plan

4.1. **Module 1: Finding the reduction matrix M.** From the endomorphism basis, we find the non-trivial idempotents and compute the matrix M. This involves computing the matrices P_{ij} from the endomorphism basis, solving the system of equations arising from $R^2 - R$ to find idempotents, selecting a suitable idempotent R, computing the integer basis for its column space using the Hermite Normal Form to find M, and verifying that M has maximal rank and satisfies the necessary invertibility condition. A wrapper function then returns M and its rank if a valid reduction is found.

The corresponding deliverable for this module is a working SageMath module that computes the matrix M and the rank m for known reducible examples and correctly reports the irreducible cases.

4.2. **Module 2: Computing the Poincaré normal form of M.** We implement the steps to transform the matrix M into the Poincaré normal form. This involves creating functions to generate the symplectic operation matrices, creating functions to apply row and symplectic operations to M and update T, implementing the Picard trick, and iteratively applying these operations to M to achieve the Poincaré normal form. The process ensures that the GCD of the minors is 1 and that $MJ^{\mathrm{T}}M$ has the required form.

By the end, we should have a working SageMath module that computes the Poincaré normal form of M and the matrix T.

4.3. **Module 3: Applying T to the period matrix.** The period matrix is reduced using the computed transformation matrix T. This involves creating a function to multiply the period matrix by T, identifying and checking the invertibility of the matrix A, applying the inverse of A to obtain the reduced period matrix, verifying the block structure of the result, and extracting the submatrices Z_1, Z_2, and Q. A wrapper function is created to return the reduced period matrix along with Z_1, Z_2, and Q.

4.4. **Module 4: Integration, Testing and Validation.** The modules are combined into a single workflow, and comprehensive tests are developed to ensure the correctness and reliability of the implementation. This involves creating a top-level function to perform reduction from a given Riemann surface X, developing test cases for each function in the preceding modules (including tests for symplectic matrix generation, elementary operations, the Picard trick, Hermite Normal Form-based basis extraction for M, block extraction, and idempotent detection), and using test cases, such as those in Bolza [1] for genus 2 curves, to verify that the computed matrices Z_1 and Z_2 are period matrices of the factor Jacobians and that the reduction achieves the form described by Martens. The outcome of this module is a fully tested implementation that correctly performs Poincaré reduction, accurately identifies decomposable Jacobians, and produces the expected reduced period matrices.

4.5. **Module 5: Documentation and Finalisation.** The code is documented and prepared for delivery. This includes writing clear docstrings for all public functions, explaining inputs, outputs, algorithms, and potential exceptions; creating project-level documentation detailing how to use the main function, dependencies, known limitations, and result interpretation; and conducting code review and refactoring for clarity and efficiency. The result is high-quality documentation that adheres to standards.

5. Timeline

The project is structured into distinct modules, each with specific goals and dependencies. The timeline below outlines the allocation of time for each module, ensuring a systematic and efficient development process. During the first four weeks, the focus will be on completing module 1, which involves finding the reduction matrix M.

In weeks 5-8, the focus will shift to module 2, which involves computing the Poincaré normal form of M by implementing the algorithm described by Martens.

The final weeks 9-12 will be dedicated to module 3 and module 4, which involve applying the transformation matrix T to the period, extracting the reduced period matrix, and integrating the modules into a cohesive workflow.

Documentation will be written concurrently with development and finalised during this period. Similarly, testing of each module will be done contemporaneously with development, ensuring that each module is thoroughly tested before moving on to the next so that no errors propagate throughout the project.

Throughout the development period, coding and development activities will be prioritised, with weekends serving as the primary periods for rest and recovery.

6. Risk Management

6.1. **Finding rational idempotents.** Finding rational idempotents can be computationally hard for Riemann surfaces of large genera or may fail for complex algebras. This is a potential risk, as the computational complexity of this task depends on the structure of the endomorphism ring. However, for the genus of curves that SageMath can reasonably handle (typically $g < 9$, where computing the Riemann matrix is not necessarily time-intensive), the dimension of the endomorphism algebra is not expected to be excessively large. Consequently, the number of variables involved in computing the Gröbner basis is likely to remain manageable.

However, to mitigate this risk, we can find the centre of A, where idempotents are more likely to lie. We will also use optimised Gröbner basis algorithms in Sage and test with known simple algebras. Additionally, we allocate buffer time if needed.

Alternatively, if this part of module 1 proves to be intractable within the timeframe, we may modify the project scope to focus on completing Modules 0, 2, 3, and 4, allowing M to be provided as input (e.g., manually derived for specific examples). We will document this limitation and suggest other ways of finding M computationally.

6.2. **Implementing the Normal Form Reduction loop.** Implementing the Normal Form Reduction loop correctly based on literature can be complex and error-prone. This complexity arises from applying a specific sequence of symplectic operations and row reductions from Martens, which must be precisely ordered to achieve the Poincaré normal form. Potential errors include incorrect operation application, miscalculating intermediate matrices, or failing to maintain the symplectic property. The algorithm also involves GCD computations and the Picard trick, further increasing complexity and error potential.

To mitigate this risk, we start with the $m = 1$ case, simplifying matrix manipulations to focus on the core reduction logic. Second, we create small, verifiable test cases for M with known normal forms to check the correctness of individual operations and the overall reduction. Finally, we conduct detailed code reviews to identify and correct logical errors, boundary conditions, and deviations from the literature, improving implementation reliability.

6.3. **Underestimation of time for testing and documentation.** In order to mitigate this risk, we start writing unit tests alongside module development simultaneously, integrate testing early and allocate dedicated time.

7. Possible extensions

If time permits, we may extend the project. We can utilise the `RiemannTheta` package and adapt it, so that we may find the factorisation of a given theta function if possible.

Most significantly, we may go from one reduction of $\mathrm{Jac}(X)$ into a complete reduction of the Jacobian, that is, $\mathrm{Jac}(X) \sim J_1 \times J_2 \times \ldots \times J_k$ where J_i are the Jacobians of lower genus curves. The research challenge here is finding an algorithm to do so. Preliminary ideas include realising that a complete reduction corresponds to the decomposition of $A = \mathrm{End}(\mathrm{Jac}(X)) \otimes \mathbb{Q}$ into a product of simple algebras. From the

decomposition of the homology group $H_1(X, \mathbb{Q})$, we must find one single symplectic basis $\{b_1, \ldots, b_{2g}\}$ for the whole lattice $\mathbb{Z}^{2g}$ that is adapted to this decomposition. The matrix T is the change-of-basis matrix from the standard basis to this new basis.

References

1. O. Bolza, *On binary sextics with linear transformations into themselves*, American Journal of Mathematics, vol. 10, no. 1, p. 47, Oct. 1887. DOI: 10.2307/2369402.
2. H. H. Martens, *A footnote to the poincaré complete reducibility theorem*, Publicacions Matemàtiques, vol. 36, pp. 111–129, Jan. 1992. DOI: 10.5565/publmat_36192_09.