Extremely Low Coverage Genomic Rearrangement Analysis with *GenCHORD*

Jack Fraser-Govil,[1★] Ronnie Crawford,[1] Max Stammnitz, Elizabeth Murchison,[2] Zemin Ning, [1]
[1] *Wellcome Sanger Institute, Hinxton, UK*
[2] *University of Cambridge, Cambridge, UK*

4 March 2025

ABSTRACT

We present *GenCHORD*, a specialised step-detection algorithm suitable for detecting large-scale copy number variations from the coverage data of a sequenced, aligned genome. This algorithm is specifically tuned for detecting Structural Variations which arise during chromothripsis-induced cancer, and so permits a high degree of data reduction, whilst preserving pertinent features for cancer classification. We describe the underlying statistical and algorithmic model, demonstrate the ability of the to preserve, compress and encode information such that a simple Machine Learning model can classify and identify subtypes of Devil Facial Tumour Disease, a cancer affecting *Sarcophilus harrisii*, the Tasmanian Devil, and discuss how this tool may be used in future.

1 INTRODUCTION

This intro (& abstract) is currently identical to the original paper. Will rewrite at a later date to avoid self-plagiarism!

The term Genomic Rearrangements refer to large-scale Structural Variations (SVs) amongst individual genomes; including large-scale deletion, insertion, duplication and translocation of megabase-length sequences of DNA, and often lead to genetic disorders (when present in the germ cells), or cancer (when arising from somatic mutations). When a Genomically Rearranged sample is sequenced and aligned against a reference the true sequence's novel adjacencies can be obscured, especially when the size of the rearranged regions are significantly larger than the read length. This is because the majority of reads will still align to the reference despite the fact that their spatial location within the genome has been altered: only those reads which bridge the unusual adjacencies contain the information that a rearrangement has occurred.

Reconstructing these rearrangements requires complex analysis, such as a haplotype phased-assembly , which is not only computationally costly, but requires the DNA be sequenced to a very high coverage, as well techniques such as Hi-C (Ijaz et al. 2024). We should like to be able to identify, analyse and classify the presence of genomic rearrangements in a simpler fashion.

To do this, we leverage the knowledge that chromothripsis-induced rearrangements are often associated with highly variable copy-number variations (Ijaz et al. 2024; Stephens et al. 2011). Where portions of the genome have been duplicated or deleted (generically termed 'gains' and 'losses'), the erroneous alignment leads to variations in the base-coverage reported by the alignment tool, as demonstrated in Figure 1. The locations of regions which have undergone gains and losses corresponds strongly to the edges of the regions which have been rearranged.

Many tools to analyse the copy-number variations exist, however in this work we modify our existing tool, **Gen**ome **C**overage **H**armonic **O**ptimiser, **R**educer and **D**enoiser (*GenCHORD*), a novel tool tailored for this problem which leverages the power of a Bayesian hypothesis testing to infer a statistically robust denoised copy-number analysis, which is suitable for encoding into a Neural Network, and hence permit accurate detection, analysis, and classification of the genomic rearrangement.

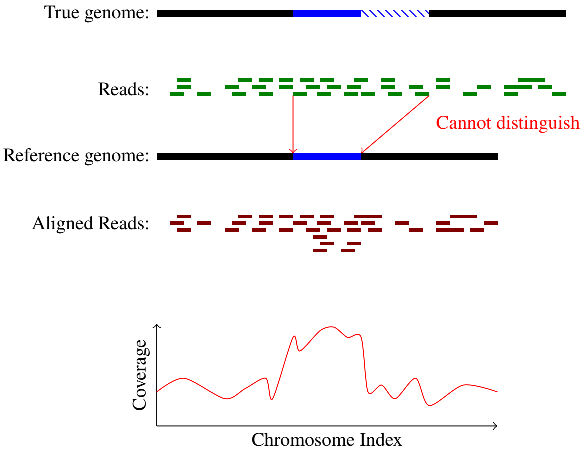

Figure 1. A depiction of duplication leading to a higher coverage. The blue hatched region is duplicated with respect to the reference, but since the reads are much shorter than the duplicated region, they manifest as a higher coverage rate.

2 GENCHORD ALGORITHM

GenCHORD is a specialised step-detection algorithm designed to leverage strong biological priors to gain a statistically meaningful dissection of the coverage variation in a genome. We presented this algorithm and demonstrated its use on high quality sequencing data, allowing us to investigate and accurately classify cancer subtypes in Tasmanian Devils (Fraser-Govil et al. 2024).

In this work, we modify the algorithm to enable it to function even on extremely low coverage datasets, as well as both long and short read platforms.

We shall present here an extremely brief analysis of the algorithm.

The full mathematical and statistical discussion can be found in appendices A, and the algorithmic concerns are discussed in appendix D.

2.1 The Data

Throughout this work we shall assume that our data is the coverage (per-base sequencing depth) extracted from an aligned genome, *GenCHORD* itself can handle data in a number of different forms, including BAM or CRAM files (via an internal call to *samtools*), as well as basic text files.

The raw data takes the form of an ordered sequence of integers, $D = \{k_1, k_2, k_3, ...\}$, where k_i is a nonnegative integer corresponding to the coverage reported by the i^{th} base in the sequence.

We shall have C such sequences, corresponding to the C distinct chromosomes present in the sample. Each sequence is of length N_c (the length of the c^{th} chromosome). Aside from the global parameter inference (appendix **??**), these sequences are analysed independently since even in a healthy sample, per-chromosome coverage can vary.

2.2 Data Aggregation

Whilst the analysis we shall present here theoretically functions on the raw, unprocessed coverage sequences, at low coverages and with noisy sequencing technologies (such as Hi-C) it can become statistically challenging to disentangle the effects of genuine genomic variation from the noise inherent in the pipeline.

For this reason, it is in fact more beneficial for us to work with *S-sums of k*. That is, we transform our data $\{k\} \to \{s\}$, such that:

$$s_i = \sum_{j=i\times S}^{(i+1)S} k_j \tag{1}$$

This is - very - subtly distinct from taking the mean of the observed coverage in a window of width S; since it is clearly true that $\bar{k}_i = \frac{s_i}{S}$. The distinction is meaningful because the mean is (often) treated as a random *continuous* variable. In fact, it is no such thing; it is discretised into increments of $\frac{1}{S}$ (if $S = 2$, it would be clear that $\bar{k}$ is a multiple of 0.5). In short, sums of random discrete variables still retain many desirable qualities which we leverage to our advantage (appendix A1.1).

Of concern for our statistical assumption is the statistical correlation between subsequent bases: if base i has coverage k, then since the read length is $\mathcal{R} \gg 1$, even with short read technologies, then we can be fairly certain that base $i+1$ will also have coverage k. Statistical variations will only likely be detected on scales $G \approx \mathcal{R}/C$, where C is the mean coverage of the genome.

We therefore recommend only incorporating every G^{th} datum into the sum, to avoid biasing the results due to this statistical dependence. If S were sufficiently large, then we would not need to take this precaution as the summation ends up 'averaging out' the bias; however we have a computational preference towards lower values of S. We therefore amend the previous equation to define our data as:

$$s_i = \sum_{j}^{S} k_{iSG+jG} \tag{2}$$

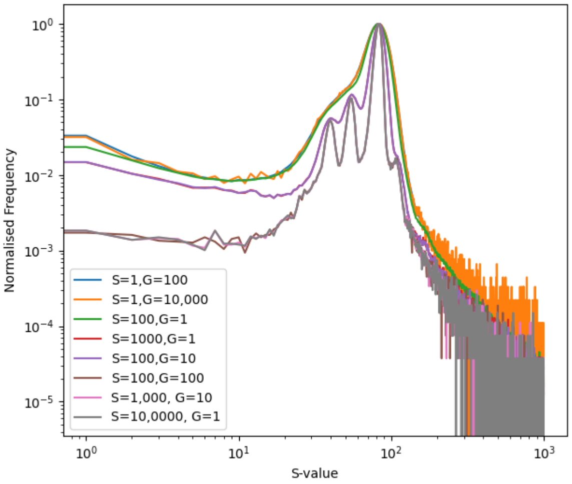

Figure 2. The difference between the raw data and aggregated data for a cancer-bearing Tasmanian Devil genome (Devil-1545T2), demonstrating the prominence of the peaks at higher aggregate values (S) and higher correlation-skipping (G). The S-value is equal to $\lfloor \frac{\sum_i^S k_{i\times G}}{S} \rfloor$, and hence equivalent (up to a binning) of the mean k observed. The y axis normalised such that the peak of every curve is at 1, for better comparison. Note that so long as $S > 100$, distributions with the same product $S \times G$ have the same pattern; indicating that we have achieved statistical independence.

Fig. 2 shows the distribution of $\lfloor \frac{s_i}{S} \rfloor$ for a variety of different values of S and G. As expected, we see that when $S > 100$, the structure of the distribution is independent of the product SG (the $S = 10,000, G = 1$ curve is visually identical to the $S = 100, G = 100$ one, for example) – this demonstrates our earlier point, that increased summation is equivalent to omitting 'correlated points' at high enough scales. However, the computational cost of computing our probability distributions is linear in the maximum observed value of s_i, which is itself a linear function of S. We would therefore prefer a lower value of S, which means a higher value of G to achieve the same data-aggregation / amplification.

2.3 Harmonic Fitting

The most obvious solution to denoise the Coverage Data is to simply pass a smoothing kernel over the data, potentially in combination with a binning algorithm, and use that to infer the underlying mean function that the data is oscillating around. Smoothing in this fashion serves as a generic way to extract a smooth curve from the extremely noisy data present, however it fails us on a number of grounds. Most importantly, this method can act to bias and manipulate the data in unforeseen and undesirable ways. Anecdotally, we found many occasions where the severity of an inferred deletion or duplication could be manipulated by altering the analysis lengthscale.

As a toy example, consider the (integer) sequence (1,5,1,1). If we we to pass a mean-kernel over this data, we would compute the mean to be $\bar{k} = 2$, which is entirely unrepresentative of the composite parts. A more involved statistical method is therefore required.

The algorithm we shall present here, is in essence is a form of *step detection*, a well known problem in signal processing. However, whilst there exist several out-of-the-box algorithms which might provide us with robust detections, we note that knowledge about the form of the data can be leveraged to provide a significantly more powerful and biologically meaningful inference.

REFERENCES

Fraser-Govil, J., Crawford, R., Stammnitz, M., Murchison, E., and Ning, Z. (2024). Genchord: A pipeline for genomic rearrangement analysis. *Proceedings of the International Conference of Computational Biology*.

Ijaz, J., Harry, E., Raine, K., Menzies, A., Beal, K., Quail, M. A., Zumalave, S., Jung, H., Coorens, T. H., Lawson, A. R., et al. (2024). Haplotype-specific assembly of shattered chromosomes in esophageal adenocarcinomas. *Cell Genomics*, 4(2).

ISO (2024). International standard iso/iec 14882:2024(e) - programming language c++.

Kingma, D. P. and Ba, J. (2017). Adam: A method for stochastic optimization.

Riley, K. F., Hobson, M. P., and Bence, S. J. (2006). *Mathematical methods for physics and engineering: a comprehensive guide*. Cambridge university press.

Stephens, P. J., Greenman, C. D., Fu, B., Yang, F., Bignell, G. R., Mudie, L. J., Pleasance, E. D., Lau, K. W., Beare, D., Stebbings, L. A., et al. (2011). Massive genomic rearrangement acquired in a single catastrophic event during cancer development. *cell*, 144(1):27–40.

APPENDIX A: MATHEMATICAL THEORY

A1 Negative Binomial

The statistical model $\mathcal{P}$ for the coverage k, is derived from the common assumption that the genome is sampled via a Poisson distribution with mean λ, but where there are biological and experimental errors which change λ, such that the model is the convolution of a Poisson distribution, and some 'error distribution', which encodes the degree to which λ varies. Maximal-entropy arguments imply that, barring any other knowledge, the Gamma distribution with mean $\mu > 0$ and variance $\sigma^2 > 0$ is the most general choice.

$$\mathcal{P} = \int_0^\infty \gamma(\lambda|\mu,\sigma)\text{Poisson}(k|\lambda)\text{d}\lambda \tag{A1}$$

This integral is analytically computable, and results in the Negative Binomial, which has a well-known form:

$$\text{NB}(k;r,p) = \frac{\Gamma(k+r)}{k!\Gamma(r)}(1-p)^k p^r \tag{A2}$$

Here $\Gamma(x)$ is the usual Gamma function with the property $\Gamma(x+1) = x!$ when x is a nonnegative integer. The resulting distribution is similar to the Poisson distribution, but 'overdispersed', with a variance which is always greater than the equivalent Poisson's: $\sigma^2_{\text{NB}} = \mu + \sigma^2$. The parameters r and p can be shown to be:

$$\begin{aligned} r(\mu,\sigma) &= \frac{\mu^2}{\sigma^2} \\ p(\mu,\sigma) &= \frac{\mu}{\mu+\sigma^2} \end{aligned} \tag{A3}$$

We emphasise again that σ^2 is the variance of the *perturbation population*, not of the Negative Binomial itself. As such, σ^2 is constrained only to be positive.

A1.1 Sums of NB-Distributions

As noted in the main text, however, we are not working with the raw coverage values, but instead with sums of coverage values; $s_i = \sum_j^S k_j$. Conveniently, the Negative Binomial also has the desirable property that it is additive (Riley et al. 2006). That is, the sum of independent Negative Binomial variables is itself distributed according to a Negative Binomial.

If k is distributed according to some $\text{NB}(r(\mu,\sigma), p(\mu,\sigma))$, then the distribution of sums of k (which we denote as s) is given by:

$$p(s|\mu,\sigma) = \frac{\Gamma(s+Sr)}{\Gamma(s+1)\Gamma(Sr)}(1-p)^s p^{Sr} \tag{A4}$$

That is, we simply replace r by $S \times r$. The key value of note for our purposes is that although this scales the mean of the distribution (predictably) by S, the standard deviation only scales as $\sqrt{S}$; the distribution therefore becomes more tightly peaked around the maximum. In the case where we are attempting to disentangle a distribution composed of multiple such overlapping distributions, this is a highly desirable property, as it will make the peaks more prominent. We see exactly this behaviour in Fig. 2.

A1.2 Normalisation

Our methods require that our models be fully normalised (i.e. the probabilities sum to 1 over the non-infinite domain of interest).

There are a number of different choices about 'when' to normalise; all of which produce a valid final final probability, but weight the internal components differently.

We choose to normalise each harmonic individually. This choice ensures that parameters such as ϵ and w_q retain their intuitive meaning as genuine probabilities. If the maximum coverage-sum permitted into the model is M, then we define:

$$B_s^q(\mu,\sigma^2) = \frac{\text{NB}(s|S\times r(\mu,\sigma), p(\mu,\sigma))}{\sum_{k=0}^M \text{NB}(k|S\times r(\mu,\sigma), p(\mu,\sigma))} \tag{A5}$$

APPENDIX B: LOCAL MODEL

The theoretical model assumes that variation in the coverage is solely a manifestation of a varying copy number, which is necessarily an integer. Each segment of the genome therefore has a coverage distribution with a mean $q\nu$, where q is the copy-number of that segment, and ν is some global mean, per-homolog coverage (in analogy to music, we term ν the 'fundamental frequency', and q values 'harmonics').

Our assertion is that, where large-scale insertions or deletions have occurred, q changes discontinuously. The statistical problem therefore reduces to inferring what q value the data has.

If we assume that the coverage is indeed distributed according to a Negative Binomial, then the probability of observing a coverage-sum s, given a known mean $\mu_q = q\nu$ and variance σ^2, error rate ϵ and error function $E(s)$ is simply:

$$p(s|q,\vec{\theta}) = \epsilon E(s) + (1-\epsilon)B_s^q(\mu_q,\sigma^2) \tag{B1}$$

Here $\vec{\theta}$ is compact notation for the parameters of the model: in this instance, $\vec{\theta} = (S,\nu,\sigma^2)$.

B1 Contamination

A crucial inclusion in the model is a means to infer *contamination*. Contamination in this sense is any confounding factor which means that q is not a perfect integer. In the simplest possible case, the sample is contaminated by a fraction η of normal somatic cells with copy number equal to the ploidy ($\mathcal{D}$); this would modify $q \to q + (\mathcal{D} - q)\eta$. The contamination value can then be provided or inferred directly.

In practice we found that more complex cancers (especially those where the cancer modified the Ploidy) were fit poorly by this model. We therefore opted for a more flexible approach, providing 'contamination parameters' for each harmonic:

$$\begin{aligned} q' &= q + \delta_q \\ \mu_q &= q'\nu \end{aligned} \tag{B2}$$

The contaminated model is therefore computed exactly as above, but replacing the Negative Binomial r and p parameters with:

$$\begin{aligned} r_q &= \frac{\left(q+\delta_q\right)^2\nu^2}{\sigma^2} \\ p_q &= \frac{\left(q+\delta_q\right)\nu}{\left(q+\delta_q\right)\nu+\sigma^2} \end{aligned} \tag{B3}$$

Although this model is more flexible, it loses the beneficial property that η - the contamination fraction - is a biologically meaningful parameter. If possible (i.e. in the case of simple cancers) we would like to recover this property. We achieve this by imposing the following ‘contamination prior’:

$$C(\{\delta_q\}, \eta) = -\mathfrak{p}_{\text{contamination}} \sum_q \left(\delta_q - (\mathcal{D} - q)\eta\right)^2 \tag{B4}$$

We retain an (approximate) estimate of the Contamination fraction; although it is now only a parameter which appears within the Prior.

B2 The Error Function

The term $E(s)$ appears in Eq. (B1) without further definition aside from being weighted to only contribute a fraction $\epsilon \ll 1$ to the total model. This is our *Error Function*, and allows us to relax our assumption that only ‘perturbed Negative Binomials’ contribute to the distribution.

For simplicity, we assume no functional form on the Error Function since it is, by definition, behaviour we have not modelled. We instead approximate this function with a piecewise-constant probability mass function with resolution R_ϵ.

$$E(s) = \begin{cases} \mathcal{E}_0 & 0 \leq s < \Delta_0 \\ \mathcal{E}_1 & \Delta_0 \leq s < \Delta_0 + \Delta_1 \\ \vdots & \\ \mathcal{E}_{R_\epsilon - 1} & M + 1 - \Delta_{R_\epsilon - 1} \leq s \leq M \end{cases} \tag{B5}$$

Where the width of each domain (Δ_i) is given by:

$$\Delta_i = \begin{cases} \left\lfloor \frac{M+1}{R_\epsilon} + \frac{1}{2} \right\rfloor & 0 \leq i < R_\epsilon - 1 \\ M + 1 - \Delta_0 (R_\epsilon - 1) & i = R_\epsilon - 1 \end{cases} \tag{B6}$$

A more compact notation can be constructed using the ‘window function’, $\Omega(k)$:

$$E(s) = \mathcal{E}_{\Omega(s)}$$
$$\Omega(s) = \begin{cases} \left\lfloor \frac{s}{\Delta_0} \right\rfloor & s < M + 1 - \Delta_{R_\epsilon - 1} \\ R_\epsilon - 1 & \text{else} \end{cases} \tag{B7}$$

We assume that the Error Function is normalised by construction and therefore satisfied:

$$\sum_{s=0}^{M} E(s) = \sum_{i=0}^{R_\epsilon - 1} \Delta_i \mathcal{E}_i = 1 \tag{B8}$$

B3 Bayes’ Theorem

For each base i, the probability of observing coverage-sum s_i, given that the current harmonic is q is therefore $p(s_i | q, \vec{\theta})$. From Bayes’ theorem, we can then invert this to find the probability $p(q | s_i, \vec{\theta})$; the probability that the harmonic is q given that we have observed s_i.

$$p(q | s_i, \vec{\theta}) = \mathcal{N} p(s_i | q, \vec{\theta}) \times \text{Prior}(q | \vec{\theta}) \tag{B9}$$

Here $\mathcal{N}$ is a multiplicative constant that depends only on the quality of the data (it is the ‘evidence’). The Prior is discussed in the next section; it is sufficient for now to note that it is in general non-separable and we must therefore consider the entire data in aggregate, rather than datapoint-by-datapoint. For a set of n data points:

$$p(\{q\} | \{s_i\}, \vec{\theta}) = \mathcal{N}^n \left(\prod_i^n p(s_i | q, \vec{\theta}) \right) \text{Prior}(\{q\} | \vec{\theta}) \tag{B10}$$

In logarithms:

$$\mathcal{L}(\{q\}) = \text{constant} + \text{LogPrior}(\{q\}) + \sum_i \log\left(p(s_i | q, \vec{\theta})\right) \tag{B11}$$

The problem then reduces to optimising this function in order to maximise $\mathcal{L}$, and therefore find the most likely set of $\{q\}$ which describes our data.

B3.1 The Prior

The *Prior* is the function enumerating our beliefs about the structure of the output before we looked at the data. For the biological problem at hand, we know that our priors should be:

(i) Consecutive values of q_i should be similar

(ii) q_i cannot alternate values rapidly. Small-scale insertions and deletions would have been resolved by the aligner.

(iii) Most of the chromosome c should be at $q_i = \mathcal{D}_c$ (equal to the ploidy)

Therefore, we propose:

$$\begin{aligned} \text{LogPrior}(\{q\}) =& \ln(\text{Prior}(\{q\})) \\ =& \underbrace{\sum_i \varphi(q_i, \mathcal{D})\, \mathfrak{p}_{\text{ploidy}}}_{\text{ploidy}} + \underbrace{\sum_i \varphi(q_i, q_{i-1})\, \mathfrak{p}_{\text{change}}}_{\text{similarity}} \\ &+ \underbrace{\sum_{i=L} \sum_{j=i-L}^{i-1} \chi(q_j, q_{i-1})\, \mathfrak{p}_\infty}_{\text{gap size}} \end{aligned} \tag{B12}$$

$$\varphi(a, b) = (1 - \delta_{ab}) = \begin{cases} 0 & a = b \\ 1 & \text{else} \end{cases}$$

The terms in the prior therefore act (in order), to penalise every base which is not at the expected diploid level by an amount $\mathfrak{p}_{\text{ploidy}}$, penalise every jump between dissimilar q by an amount $\mathfrak{p}_{\text{change}}$ and to penalise jumps which occur within a distance L of another jump by an amount $\mathfrak{p}_\infty$, which is a number $\approx -\infty$, but which has the property $0 \times \mathfrak{p}_\infty = 0$ – rather than a penalty, this therefore acts as a *forbiddance*, discarding models with such jumps.

This final term of the prior – the forbiddance – is encoded in the structure of the algorithm (such models are never considered, rather than being considered and then removed).

The total prior also includes any other priors we have imposed; for example Eq. (B4):

$$\text{LogPrior}(\vec{\theta}) = \text{LogPrior}(\{q\}) + C(\{\delta_q\}) \tag{B13}$$

APPENDIX C: PARAMETER INFERENCE & THE GLOBAL MODEL

The ‘local’ model discussed above is sufficient to infer the values of q each base is to be assigned. However, it relies on a large number of model parameters: the fundamental frequency ν, the variance σ^2, the contamination parameters $\{\delta_q\}$, the error rate ϵ and the components of the residual function $E(s)$. Before we can assign $\{q\}$ we must infer these values. We could place them into the same inference routine as $\{q\}$, except that this requires genome-scale computations, and so rapid iteration and optimisation is not feasible.

Instead, we aggregate the data into a histogram; rather than worrying about *where* each q is, we instead attempt to compute *how many there are*. Taking our data $D = \{s_1, s_2, \ldots\}$, we transform it into a vector $\vec{N}$ such that N_r is the number of times that $s = r$ occurs in the data. For convenience, we also truncate the data at some maximum value, M. This prevents wasting computation time on extreme outliers. M can usually be selected by ensuring that 99% of the data lies below the value.

C1 The Global Model

The Global Model differs from the Local Model insofar as, rather than trying to compute where two populations are separated, we model the genome as a mixture model of all the harmonics given in Eq. (B3):

$$\begin{aligned} p_{\rm global}(s|\vec{\theta}) &= \sum_{q=0}^{Q} w_q p(s|q,\vec{\theta}) \\ &= \epsilon E(s) + (1-\epsilon)\sum_{q=0}^{Q} w_q B_s^q \end{aligned} \tag{C1}$$

Due to our choice of normalisation in Eq. (A5), w_q are the weights of each harmonic in the mixture model; equal to the fraction of the genome assigned to each q. However, unlike in the local model, we have no idea *where* those q values are. The w_q are interpreted as probabilities (i.e. the probability of belonging to members of the mixture), and thus are constrained such that $w_q \geq 0$ and $\sum_q w_q = 1$.

If this condition on w_q holds, and both $E(s)$ and B_s^q are independently normalised on the domain $[0, M]$, it is also guaranteed that $\sum_s p_s = 1$. With this normalisation, we can use Bayes' theorem to set up a likelihood:

$$\mathcal{L}_{\rm global}(\vec{\theta}) = \sum_{s=0}^{M} N_s \log\left(p_s(\vec{\theta})\right) + {\rm LogPrior}(\vec{\theta}) \tag{C2}$$

C2 Limitations of Separable Optimisation

By utilising the Global Model to infer $\vec{\theta}$, and then using this to infer $\{q\}$ in the Local Model, we are performing a *separable optimisation*. For full mathematical validity, we should optimise these *simultaneously*. However, this is computationally prohibitive.

There do exist pathological cases where the separable-optima is distinct from the simultaneous-optima. These cases arise where the data oscillates q more rapidly than the minimum jump size. The Global Model has no sense of separation (relying solely on counts), and so would assign ν based on the 'true' average; whilst a simultaneous would attempt to compromise the average by minimising the number of jumps.

In the case of real data, however, the separable optimisation will return results close to the simultaneous optimisation.

C3 Optimisation

Seeking the optimum value of of the parameters, $\vec{\theta}$ is made complex by the large number of local optima: it is clear that if the true maxima lies at $\nu = 30$, then $\nu = 15$ will also exhibit a sharp peak, since (by design), the model is sensitive to integer multiples of the harmonic frequency.

A naive optimisation algorithm is therefore highly likely to get caught in a local optimum.

We therefore utilise an algorithm which uses simulated annealing techniques to locate the approximate region of an optima, before using traditional gradient based techniques to converge rapidly onto the optima.

C3.1 Annealed-Optimisation

We use standard simulated annealing and Markov-Chain Monte Carlo techniques. Starting from a 'position', $\vec{\theta}_0$, we generate a second 'proposed' position, $\vec{\theta}_1$. The two positions are assigned an 'energy' using Eq. (C2):

$$\Delta E(\vec{\theta}_i, \vec{\theta}_j) = \mathcal{L}_{\rm global}(\vec{\theta}_j) - \mathcal{L}_{\rm global}(\vec{\theta}_i) \tag{C3}$$

The probability of accepting the proposed position is then determined by the 'temperature' of the model:

$$p_{\rm accept}(\vec{\theta}_j|\vec{\theta}_i) = \min\left(1, \exp\left(\frac{\Delta E(\vec{\theta}_i, \vec{\theta}_j)}{T}\right)\right) \tag{C4}$$

When $T \gg \Delta E$, we are likely to accept proposed positions even if they are significantly worse than our current position; otherwise the update is rejected, and we generate a new candidate position $\vec{\theta}_{j+1}$. By iterating this procedure many times, we allow the model to move between local peaks.

Of crucial importance is controlling the the value of T; cooling it over time to force the model to converge to the optima, but we must also allow T to increase again if the acceptance fraction (i.e. the number of proposed positions which have been accepted) is too small. By repeatedly 'heating and cooling' the system, we allow the model to widely explore the parameter space, and avoid getting caught in minima.

C3.2 Proposed Positions

We initially follow the standard Metropolis-Hastings prescription for proposing a new position: the proposed positions are generated using a multivariate Gaussian centred at the previous position, with standard deviation d. Proposed updates which are close to the previous are more likely than those which are far away; but large steps are possible.

$$\vec{\theta}_{i+1} \sim \mathcal{N}\left(\vec{\theta}_i, d\right) \tag{C5}$$

However, as the model cools, we introduce a second method of proposing a position: by performing a number of steps of a standard Gradient descent algorithm (the ADAM routine of (Kingma and Ba 2017))

$$\vec{\theta}_{i+1} = {\rm ADAM}(\vec{\theta}_i) \tag{C6}$$

As the temperature 'cools', we perform optimisation for longer, allowing more accurate estimations of the current optima value.

C3.3 The Derivatives

In order to perform Gradient Descent Optimisation, we must compute:

$$\frac{\partial \mathcal{L}_{\rm global}}{\partial \vec{\theta}} = \sum_{s}^{M} \frac{N_s}{p_s} \frac{\partial p_{\rm global}(s,\vec{\theta})}{\partial \vec{\theta}} + \frac{\partial {\rm LogPrior}}{\partial \vec{\theta}} \tag{C7}$$

This is not a particularly conceptually challenging task, but due to the number of components of the model, it can get rather convoluted.

We begin with introducing a number of auxiliary variables. These enable us to perform an unconstrained optimisation, whilst ensuring the constraints of our parameters are met:

$$\nu = \exp(x) \qquad \text{Frequency always positive} \tag{C8}$$

$$\sigma^2 = \exp(y) \qquad \text{Variance always positive} \tag{C9}$$

$$w_q = \frac{\exp(z_q)}{\sum_r \exp(z_r)} \qquad w_q \text{ always positive and sum to 1} \tag{C10}$$

$$\epsilon = \frac{e}{1+\exp(-\phi)} \qquad \text{Error rate always between 0 and } e \tag{C11}$$

$$\mathcal{E}_i = \frac{\exp(\rho_i)}{\sum_{j=0}^{R_\epsilon - 1} \Delta_j \exp(\rho_j)} \qquad \text{Error function normalised on } [0, M] \tag{C12}$$

$$\delta_q = d_- + \frac{d_+ - d_-}{1+\exp(-\psi_q)} \qquad \text{Contamination always between } d_- \text{ and } d_+ \tag{C13}$$

Our optimisation vector is therefore $\vec{\theta} = (x, y, \{z_q\}, \phi\{\rho_i\}, \{\psi_q\})$, where the number of z_q and δ_q components is equal to Q (the 'harmonic level') and the number of ρ_i is equal to R (the 'error resolution'), all of which are user-provided values. We then define the following intermediary values:

$$\begin{aligned} \langle s\rangle_s &= \sum_s s B_s^q \\ \langle \mathsf{F}\rangle_q &= \sum_s B_s^q \mathsf{F}(s + Sr_q) \\ \tau_s^q &= Sr_q\left(\mathsf{F}(s+Sr_q) - \langle \mathsf{F}_q\rangle\right) \\ \lambda_s^q &= p_q\left(\langle s_q\rangle - s\right) \end{aligned} \tag{C14}$$

Where $\mathsf{F}(x)$ is the (first-order) digamma function. This allows us to write the various derivatives as:

Fundamental Frequency

$$\frac{\partial p_{\rm global}(s)}{\partial x} = (1-\epsilon)\sum_q w_q B_s^q \left(2\tau_s^q + \lambda_s^q\right) \tag{C15}$$

Variance

$$\frac{\partial p_{\text{global}}(s)}{\partial y} = -(1-\epsilon)\sum_q w_q B_s^q \left(\tau_s^q + \lambda_s^q\right) \tag{C16}$$

Weights

$$\frac{\partial p_{\text{global}}(s)}{\partial z_q} = w_q \left(B_s^q - \sum_r w_r B_r^q\right) \tag{C17}$$

Contamination

$$\begin{aligned}\frac{\partial p_{\text{global}}(s)}{\partial \psi_q} &= (1-\epsilon) w_q B_s^q \left(2\tau_s^q + \lambda_s^q\right) \\ &\times \frac{(\delta_q - d_-)(d_+ - \delta_q)}{(d_+ - d_-)(q + \delta_q)}\end{aligned} \tag{C18}$$

Error Rate

$$\frac{\partial p_{\text{global}}(s)}{\partial \phi} = \epsilon\left(1 - \frac{\epsilon}{e}\right)\left(E(s) - \sum_q^Q w_q B_s^q\right) \tag{C19}$$

Error Function

$$\frac{\partial p_{\text{global}}(s)}{\partial \rho_j} = \epsilon \mathcal{E}_{\Omega(s)}\left(\delta_{j,\Omega(s)} - \mathcal{E}_j \Delta_j\right) \tag{C20}$$

(Note here that we are using $\delta_{a,b}$ in the sense of the Kroencker Delta, not the contamination parameter, δ_q)

APPENDIX D: GENCHORD ALGORITHM

D1 Gamma-Digamma Simplification

The nature of the Negative Binomial requires that we be able to compute the following (transcendental) functions of the Gamma function (Γ) and its derivative (F):

$$\begin{aligned} f(s) &= \log\left(\frac{\Gamma(s+r)}{\Gamma(r)}\right) \\ h(s) &= F(s+r) - F(r)\end{aligned} \tag{D1}$$

There exist efficient methods to approximate these functions (for example, the C++ standard (ISO 2024) guarantees the existence of `std::lgamma` to compute $\log(\Gamma(x))$).

Our uses, however, require being able to compute these functions for all values of s, sequentially, on the domain $[0, M]$. This is a considerable source of computational expenditure, and can be vastly reduced by the use of the following identities, leveraging our knowledge that $s \in \mathbb{N}$ and $r > 0$:

$$\begin{aligned} f(s) &= \begin{cases} 0 & s = 0 \\ f(s-1) + \log(s-1+r) & \text{else}\end{cases} \\ h(s) &= \begin{cases} 0 & s = 0 \\ h(s-1) + \frac{1}{s-1+r} & \text{else}\end{cases}\end{aligned} \tag{D2}$$

Although slower for computing *individual* values of $h(s)$ and $f(s)$, this recurrence relation allows us to compute *sequential* values with vastly reduced overhead: replacing expensive Γ and F calls with (comparatively) cheap division and logarithms. This can result in speedups in excess of 10x when S is large.