Deforesting Coverage Data

Jack Fraser-Govil

May 21, 2024

1 INTRODUCTION & MOTIVATION

Coverage data belies a wealth of data – we have already seen that a hidden set of biases can be seen when considering the coverage-frequency distribution.

In addition, Coverage can also be used to detect the presence of large-scale copy-number variations and other forms of duplication or deletion (which we shall generically call 'gains' and 'losses'). In the case of losses it is easy to see why this would be the case, it is perhaps less obvious why it happens in the case of gains.

Provided the read length is smaller than the scale of the duplications we are interested in, it is generally not possible to infer which copy of a motif a read originated from, and so the alignment does not record this as an insertion/duplication of bases – it simply assigns them to the original copy in the reference, leading to a spurious over-coverage of the reference, as shown in Figure 1.

Only reads which bridge the gap between consecutive copies would permit us to properly infer the presence of duplication – however, if the number of duplicates is greater than two, then there are multiple such bridge points, and the problem is highly degenerate and we must rely on the coverage to infer the number of duplications.

The problem with this kind of inference is that Coverage is already inherently stochastic and highly noisy - coverage plots covering any significant portion of the genome are difficult to interpret due to a large degree of shot noise which – as we have seen – has a significantly higher dispersion than Poisson shot noise, with the noise changing on (approximately) a per-base resolution.

Plots of the base coverage are - to be charitable - hard to interpret, reminiscent of the notorious 'Lyman-Alpha forest' in astrophysical spectroscopy. Whilst large-scale trends can (just about) be picked out by eye, we might desire a more rigorous and sophisticated methodology.

The aim of this work is to generate a computational tool which can peer through the 'Forest', and see the underlying pattern.

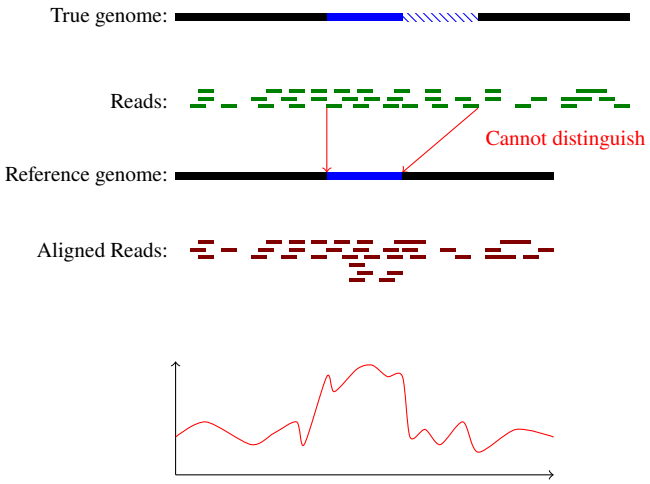

Figure 1. A depiction of duplication leading to a higher coverage. The blue hatched region is duplicated with respect to the reference, but since the reads are much shorter than the duplicated region, they cannot be properly assigned as duplicates, instead leading to a higher coverage rate.

2 MATHEMATICAL THEORY

2.1 Naive Smoothing

The most obvious solution is to simply pass a smoothing kernel over the data, and use that to infer the underlying mean function that the data is oscillating around. The Nadarya-Watson (cite?) method uses a kernel function $K_\lambda(x, y) = \frac{1}{\lambda} k\left(\frac{|x-y|}{\lambda}\right)$, and the mean estimate of a set of data (x_i, y_i) is given by:

$$\hat{m}_\lambda(x) = \frac{\sum_i^n K_\lambda(x, x_i) y_i}{\sum_i K_\lambda(x, x_i)} \tag{1}$$

The kernel should be sufficiently decaying that $K_\lambda(x, y) = 0$ when $|x - y| \gg \lambda$, where λ is a characteristic 'smoothing lengthscale'.

Smoothing in this fashion serves as a generic way to extract a smooth curve from the extremely noisy data present (i.e. the 'forest') - however it fails us on a number of grounds:

- It is not biologically motivated (no underlying mechanism present), without regard to any other information we have about the system.
- It reduces the coverage to a smooth curve. This is undesirable because:
  - It gives us no indication about where the gains and losses begin or end (the quantity of real interest to us) – we would need to do some more statistical post-processing in order to infer the edges of the transitions. The smoothing only helps human eyes see things, it does not particularly help in our numerical affairs.
  - The action of the kernel is (by definition) to smooth out sharp transitions. However – when a transition occurs we *expect* it to be a sharp transition. The kernel method necessarily smooths out these edges – it is difficult to distinguish between 'noise' and 'genuine transitions'.

2.2 Harmonic Fitting

The primary problem with the naïve smoothing approach was that it (by definition) *smoothed out* the data, in order that we might more easily spot *discontinuities* in the data. These two points – smoothing and discontinuities – stand in direct opposition to each other, and so the method is inherently flawed.

We should therefore attempt to identify the discontinuities straight from the dataset. This, 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.

2.2.1 Model Assumptions

We use the following knowledge and assertions as the underpinnings of our model:

(i) The distribution of coverage, k, is approximately Poisson-like, around some central value λ

(ii) The value of λ is constant across a chromosome, except during gains or losses, where it changes discontinuously

(iii) The gains and losses occur throughout the sample (i.e. no subclonality)

- As a result, duplication or deletion of a region results in λ changing by integer multiples ('harmonics', denoted as q) of an underlying 'fundamental frequency', ν.
- Most of the sample should have a harmonic of $q = 2$ - normal human diploidy.

(iv) Discontinuities must be separated by at least a distance L on the linear genome, else they would have been resolved by the alignment.

(v) There is some error in the alignment method[1], such that a base assigned coverage k_{obs} actually had coverage k_{true}.

(vi) We should prefer a model with fewer jumps to one with more, all else being equal.

The key assumption here is iii: the assumption that changes in coverage occur in **integer steps** of the fundamental frequency ν, such that the average coverage in a region is always $q\nu$, where q is our integer 'harmonic'. It is this distinction which makes our step-detection algorithm highly distinct from other methods.

It is worth noting that we are only able to detect the *absolute* harmonic, rather than the relative: Motifs which have multiplicity in the reference (such as in the centromere or telomere) will show up as high-q values, and low q values will appear where mis-alignment has occurred. Therefore motifs which are already repetitive but which have suffered gains or losses will move from one q to another – we will only be able to detect these regions by comparing to a healthy sample.

The task of detecting the 'steps', therefore, requires that we assign each base a value of q: gains and losses are trivially detectable wherever this integer value changes.

2.2.2 Statistical Model

In order to assign the values of $\{q\}$ to a set of genome coverage data $\{k\}$, we must find a way of assigning a statistical score, $\mathcal{L}$ to a proposed set of harmonics, $\{q\}$. The task of step-detection then reduces to finding the value of $\{q\}$ which maximises $\mathcal{L}$, and is therefore the most likely given the observed data.

We make the standard assumption that the coverage k_i of the i^{th} base is drawn independently and identically from an underlying distribution function (*iid*), which is a function of the fundamental frequency, ν, and the harmonic, q_i. Bases are statistically connected through the *prior* function, which encodes many of the points outlined in the previous section.

As a result of the *iid* assumption, the global score can be found to be:

$$\mathcal{L}(\{q\}|\{k\},\nu) = \sum_i \mathfrak{p}\left(k_i|q_i,\nu\right) + \text{Prior}(\{q\}) \tag{2}$$

This is an indirect statement of standard Bayesian Maximum Posterior methodology - the quantity $\mathfrak{p}$ is the log-probability of observing the data k_i given the harmonic q_i and parameter ν.

The probability $p(k_i|q_i,nu)$ can be formulated from our assumptions - namely that we are looking for an 'error prone', Poisson-like distribution. There are two possible sources of additional noise which we handle slightly differently. The first source of noise is an inherent deviation from the underlying Poisson assumption - this follows from our previous work that the distribution is highly non-Poisson for a variety of reasons: we term this *process noise*, and it is a fundamental part of the underlying biology. .

The process noise we model by assuming that the underlying Probability model is, instead of being a pure-Poisson, marginalised against a Gamma distribution with mean μ and variance σ^2 (in the common parameterisation this is $(\alpha,\beta) = \left(\frac{\mu^2}{\sigma^2}, \frac{\mu}{\sigma^2}\right)$):

$$\begin{aligned} p(k|\mu,\sigma) &= \int_0^\infty p(k|\lambda)p(\lambda|\mu,\sigma)\mathrm{d}\lambda \\ &= NB\left(k; \frac{\mu^2}{\sigma^2}, \frac{\mu}{\mu+\sigma^2}\right) \\ &\left(NB(k,r,p) = \frac{\Gamma(k+r)}{k!\Gamma(r)}(1-p)^k p^r\right) \end{aligned} \tag{3}$$

Where $NB(k;n,r)$ is the usual Negative Binomial distribution, written in terms of Real (rather than integer) r, and $\Gamma(x)$ is the usual Gamma function. The Harmonic assumption is embedded in the assumption tha we will always use $\mu_i = q\nu$, where q is an integer.

The second source is *experimental noise* - contamination from different cell populations, misalignments and so on. From empirical observation of the underlying distribution function, we make the ansatz that there is another NB distribution which generates coverage values with a mean μ_e and a variance $\sigma_e > \sigma$.

$$p_{\text{noise}}(k|\mu_e,\sigma_e) = NB\left(k; \frac{\mu_e^2}{\sigma_e^2}, \frac{\mu_e}{\mu_e+\sigma_e^2}\right) \tag{4}$$

The probability that a given base with observed coverage k was generated by the q^{th} harmonic is then equal to:

$$\begin{aligned} p_{\vec{\theta}}(k|q) &= p(k|q,\nu,\sigma,\gamma,\mu_e,\sigma_e) \\ &= (1-\gamma)NB\left(k; \frac{(q\nu)^2}{\sigma^2}, \frac{q\nu}{q\nu+\sigma^2}\right) + \gamma NB\left(k; \frac{\mu_e^2}{\sigma_e^2}, \frac{\mu_e}{\mu_e+\sigma_e^2}\right) \end{aligned} \tag{5}$$

Figure 2 shows some examples of this distribution (the 'Errored-Binomial') compared to the basic Poisson Distribution and the Negative Binomial distribution, demonstrating its desirable properties.

Finally, we must formulate a suitable prior, which enforces the remainder of our assumptions, namely:

(i) Consecutive values of q_i should be similar
(ii) Changes in q_i can only happen over a large distance
(iii) Most of the genome should be at $q_i = 2$ (equal to the ploidy)

[1] This is most important in regions of deletion, since $\lambda = 0$ is very difficult to infer if even a single read is misaligned into a deleted region

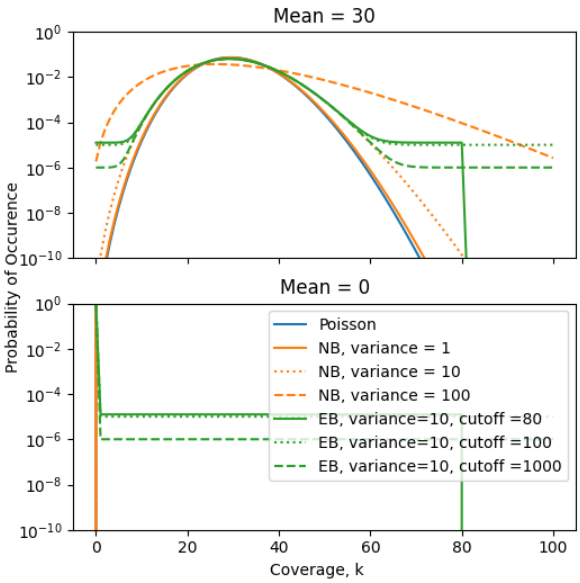

Figure 2. Various distribution functions: Poisson, Negative Binomial (NB) and our 'Errored-Binomial' (EB) are shown with various parameters, and $\mu = 30$ (top panel) and $\mu = 0$ (lower panel). The key feature of the EB distribution is that it resembles the NB distribution near the peak, but has wider wings – most notably when $\mu = 0$ (corresponding to total deletion), where the Poisson and NB distributions are zero everywhere except $k = 0$, whilst the EB distribution permits non-zero values. As a result, the EB-distribution has a mean slightly greater than zero – μ refers to the 'true sampling' mean, rather than the observed.

Therefore, we propose:

$$\text{Prior}(\{q\}) = \sum_i \left[\mathfrak{p}_{\text{ploidy}} u(q_i, 2) + u(q_i, q_{i-1}) \left(\mathfrak{p}_{\text{jump}} + \mathfrak{p}_\infty \sum_{j=i-L}^{i-1} u(q_j, q_{i-1}) \right) \right] \tag{6}$$

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

The terms in the prior therefore act (in order), to penalise every base which is not at the expected diploid level by an amount $\mathfrak{p}_{\text{diploid}}$, penalise every jump between dissimilar qs by an amount $\mathfrak{p}_{\text{jump}}$ and to penalise jumps which occur within a distance L of another jump by an amount $\mathfrak{p}_\infty$, which is a number[2] $\approx -\infty$, 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.

2.2.3 *Algorithmic Concerns*

Equations (**??**) and (6) provide everything we need to compute the score Eq. (2) for a proposed set of $\{q\}$. We therefore now need an algorithm which provides the maximal set (though, as is common in such endeavours, we instead attempt to find the set which minimises $-\mathcal{L}$, an equivalent task, since $\mathcal{L} < 0$ everywhere).

[2] A surreal number, in fact

Provided that we are able to make the assumption that there is some finite Q such that $0 \leq q < Q$ across the entire genome, it is most convenient to think in terms of a *network*, such that an assignment of $\{q\}$ to corresponds to a unique path through the network. Finding the optimal assignment is therefore equivalent to finding the shortest path through this network.

The network is arranged in a series of layers (not dissimilar in structure to the network diagrams of feedforward neural networks, whence this idea was derived) - each layer corresponding to a given base in the genome and its associated coverage value. The vertices (or 'nodes', to continue the FNN analogy) in the layer then correspond to different values of q that might be assigned to that base. In short, a node n_{iq} corresponds to the i^{b} base being assigned a harmonic of q.

The edges between vertices are directional (so "a is connected to b" means that $a \rightarrow b$ is possible, but not $b \rightarrow a$), and obey the following rules, which are displayed graphically in Figure 3:

(i) The start node is connected to all nodes in the first layer, n_{0q} $0 \leq q < Q$
(ii) The node n_{iq} is connected to $n_{i+1,q}$ ('step edges') and $n_{i+L,j}$ ('jump edges') where $j \neq q$ (provided that $i + L < G$, the genome size – this prevents 'jumping off the edge' of the chromosome)
(iii) The nodes $n_{G-1,q}$ are connected to the End node

Each step edge (including those from the start node) has a cost:

$$\text{Cost}(n_{iq} \rightarrow n_{i+1,q}) = p_{\vec{\theta}}(k_i|q) + \mathfrak{p}_{\text{ploidy}} u(q, 2) \tag{7}$$

A 'step-edge' therefore corresponds simply to assigning the associated q to that base/layer.

The 'Jump-edges' from $n_{iq} \rightarrow n_{i+L,j\neq q}$ enforce the requirement that any jumps must be larger than L, since they are the only edges which connect different q values. Traversing along a jump edge corresponds to assigning all nodes $i + 1 \rightarrow i + L$ to the harmonic $j \neq q$, and hence has an associated cost:

$$\text{Cost}(n_{iq} \rightarrow n_{i+L,j\neq q}) = \mathfrak{p}_{\text{jump}} + \sum_{a=i+1}^{i+L} \left(p_{\vec{\theta}}(k_i|j) + \mathfrak{p}_{\text{ploidy}} u(j, 2) \right) \tag{8}$$

Here we have used the shorthand $p_{\vec{\theta}}(k|q) = p(k|q\nu, \sigma, \gamma, K)$, our underlying 'Errored Binomial' distribution function.

From these rules it is a simple matter of using a basic shortest-path algorithm: step through each layer and compute the shortest path to each node in that layer, and then iterate until the end node is reached – the shortest path through this network corresponds to a unique $\{q\}$, which is then necessarily the one which minimises $\mathcal{L}$. An example of such a minimal path is shown in Fig. 4.

The runtime of this algorithm scales as $O(Q^2 G)$, where Q is the maximum harmonic and G is the number of bases being evaluated. However, if we naively construct this network in-memory it occupies a space mQG, where m is the memory footprint of an individual node. Assuming that $Q = 20$ and each node must know its score, q-value and a pointer to the previous node, this gives a conservative estimate of 646GB of active memory to construct a network for the entire human genome – well beyond what most consumer devices are capable of.

However, we note that this network is 'semi-local' – by design, the edges never extend beyond a distance L from the current base. Therefore we only need LQ nodes in memory at once, rather than GQ, a drastic reduction in memory footprint: $L = 10^5$ reduces the memory footprint to only 34MB.

Each node therefore exists on a 'conveyor belt', an array of length

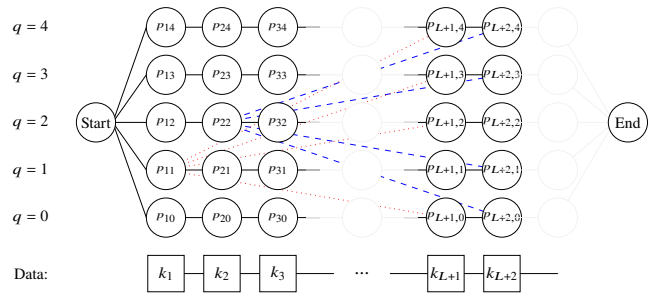

Figure 3. An example of a harmonic network – only two sets of 'jump edges' are shown (in red and blue) for clarity. In the full network, every node p_{iq} is connected to $p_{i+1,q}$ and $p_{i+L,k\neq q}$. The graph is directional - vertices can only be traversed from left to right.

L+1, with node n_{iq} located at index $C_{i\%(L+1),q}$. The step-connected nodes are at $C_{(i-1)\%(L+1),q}$ and the jump connected nodes at $C_{(i+1)\%(L+1),q}$, where $a\%b$ is the usual modular remainder operation. This means that (aside from the 'edges' of the conveyor belt), there is remarkable data contiguity.

The major drawback of this implementation is that (since the optimal path cannot be retraced by simply backtracking through each node's 'best path' pointers) each node needs to keep a record of its entire shortest path. We ameliorate this concern slightly since we know that most of the path will (by design) be through step-edges, and therefore only need to record and keep the limited number of jump edges – however, passing these records around is still the limiting factor of this algorithm.

However, we note that during the search for the value of ν (during which time many paths are evaluated, searching for the one which minimises $\mathcal{L}$), we only need the final 'length' of the path, not the path itself. For the bulk of function calls to the path-tracing algorithm, therefore, we do not need to record the path - only its length – and hence the runtime difference between the conveyor-belt algorithm and the naive algorithm is negligible, with the memory-footprint nevertheless vastly reduced.

APPENDIX A: PARAMETER OPTIMISATION

Although it would be possible to infer the values of $\nu, \sigma, \gamma, \mu_e, \sigma_e$ by optimising them against Eq. (2), this is computationally quite intensive - since it involves multiple passes through the network to explore this multidimensional parameter space. It is therefore much easier to fit these global parameters to the *global distribution* of coverages, before then using them to infer the *local* values of q_i.

If we truncate the data to only look at k up to some maximum value K, the global distribution function is equal to:

$$\begin{aligned} p_{\text{global}}(k|\vec{\theta}) &= \sum_q w_q p_{\vec{\theta}}(k|q,K) \\ &= \sum_q w_q \frac{p_{\vec{\theta}}(k|q)}{\sum_{k'=0}^{K} p_{\vec{\theta}}(k|q)} \end{aligned} \tag{A1}$$

Where w_q are the weights associated with each of the subpopulations; rather than assignined each base to a given q were are fitting them as an aggregate population. The Bayesian score associated with data binned such that N_k is the number of bases assigned the coverage k

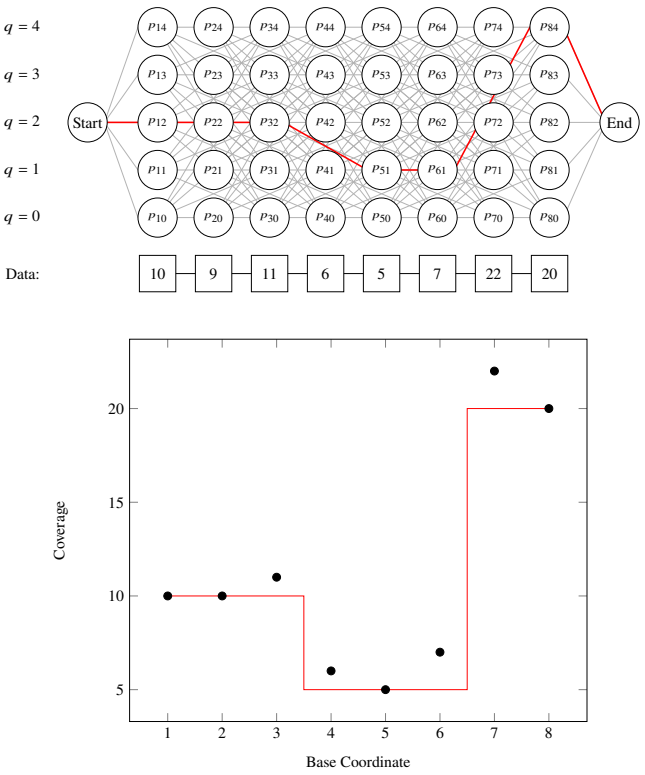

Figure 4. (Top panel) A demonstration of an optimal path through a network with $L = 2$ and $\nu = 5$, given some example coverage data. (Bottom panel) a projection of this path back onto the coverage distribution. For aesthetic reasons we have placed the transitions at half-integers – in practice non-integer values of base index are meaningless.

is then:

$$\mathcal{L}_{\text{global}} = \text{LogPrior}(\vec{\theta}) + \sum_k N_k \ln\left(p_{\text{global}}(k|\vec{\theta})\right) \tag{A2}$$

$$= \text{LogPrior}(\vec{\theta}) + \sum_k N_k \ln\left(p_{\vec{\theta}}(k|q)\right) - N \ln\left(\sum_k p_{\vec{\theta}}(k|q)\right) \tag{A3}$$

Where $N = \sum_k N_k$ is the total number of observations. The parameters are then optimised using the gradients:

$$\frac{\partial \mathcal{L}}{\partial \theta_i} = \frac{\partial \text{LogPrior}(\vec{\theta})}{\partial \theta_i} + \sum_k \left(\frac{N_K}{p_k} - \frac{N}{\sum_k p_k}\right) \frac{\partial p_k}{\partial \theta_i} \tag{A4}$$

Where (using the notation that $B_{kq} = NB\left(k; \frac{(q\nu)^2}{\sigma^2}, \frac{q\nu}{q\nu+\sigma^2}\right)$, $E_k =$

$$NB\left(k;\frac{\mu_e^2}{\sigma_e^2}\right), r_e=\frac{\mu_e^2}{\sigma_e^2}, p_e=\frac{\mu_e}{\mu_e+\sigma_e^2}, r_q=\frac{\nu^2q^2}{\sigma^2} \text{ and } p_q=\frac{q\nu}{q\nu+\sigma^2})$$

$$\begin{aligned}
\frac{\partial p_k}{\partial \gamma} &= E_k - \sum_q w_q B_{kq} \\
\frac{\partial p_k}{\partial z_j} &= (1-\gamma)w_j\left[B_{kj} - \sum_q w_q B_{kq}\right] \\
\frac{\partial p_k}{\partial \mu_e} &= \gamma E_k\Big[\frac{2\mu_e}{\sigma_e^2}\left(\psi\left(k+r_e\right)+\ln\left(p_e\right)-\psi\left(r_e\right)\right) \\
&\quad -\frac{\sigma_e^2}{(\mu_e+\sigma_e^2)^2}\frac{p_e(k+r_e)-r_e}{p_e^2}\Big] \\
\frac{\partial p_e}{\partial \sigma_e} &= 2\gamma E_k\Big[\frac{\sigma_e}{(\mu_e+\sigma_e^2)^2}\frac{p_e(k+r_e)-r_e}{p_e^2} \\
&\quad -2\frac{\mu_e^2}{\sigma_e^3}\left(\psi\left(k+r_e\right)+\ln\left(p_e\right)-\psi\left(r_e\right)\right)\Big]
\end{aligned} \tag{A5}$$

$$\begin{aligned}
\frac{\partial p_e}{\partial \nu} &= (1-\gamma)\sum_q w_q B_{kq}\Big[\frac{2\nu q^2}{\sigma^2}\left(\psi\left(k+r_q\right)+\ln\left(p_q\right)-\psi\left(r_q\right)\right) \\
&\quad -\frac{\sigma^2}{(q\nu+\sigma^2)^2}\frac{p_q(k+r_q)-r_q}{p_q^2}\Big] \\
\frac{\partial p_e}{\partial \sigma} &= 2(1-\gamma)\sum_q w_q B_{kq}\Big[\frac{\sigma}{(q\nu+\sigma^2)^2}\frac{p_q(k+r_q)-r_q}{p_q^2} \\
&\quad -2\frac{q^2\nu^2}{\sigma^3}\left(\psi\left(k+r_q\right)+\ln\left(p_q\right)-\psi\left(r_q\right)\right)\Big]
\end{aligned} \tag{A6}$$