The CBC template-bank mass model: what `sgn-manifold-cbc-bank-mass-model` computes

Draft notes for discussion compiled by Chad Hanna and Claude Fable 5.1

September 10, 2026

Abstract

These notes document the mathematics implemented by the bank mass-model tool and the library code it calls. The tool takes an astrophysical population model α and discretises it onto the template bank: $P(\theta_j \mid \alpha)$ is the probability that a source drawn from α is represented by the parameters θ_j at the centre of template j's cell. It then propagates that discretised population through a recovery kernel $P(t_k \mid \theta_j, \rho)$, the probability that a signal with parameters θ_j observed at matched-filter SNR ρ is recovered by template t_k, to obtain the per-template, per-SNR recovery probability $P(t_k \mid \alpha, \rho)$. The result is stored as a cubic spline in ρ for every template and consumed by the ranking statistic. The stored model describes the *intrinsic* population; SGNL adds the selection effect of the detector's reach on the fly. The file also carries per-template horizon distances and two population-mass scalars, from which the p_{astro} calculation in this repository forms the corresponding detected-population quantities.

Contents

1 Overview

The command-line script, `src/sgn_manifold/bin/manifold_cbc_bank_mass_model.py`, is a driver. It loads a bank, validates the model specification, and runs four stages, each implemented in `src/sgn_manifold/sources/cbc.py` unless noted:

1. the *discretised intrinsic population* $P(\theta_j \mid \alpha)$, the code's "template prior", through the cell masses μ_j (`CBCCoordFunc._power_law`, plus `mass_model_volume`, `_population_density` and `_spin_density` on the coordinate function);
2. the *horizon data* that convert intrinsic quantities to detected ones: per-template horizon distances and the canonical BNS horizon (`Bank.template_horizons`);
3. the *recovery model* $\log P(t_k \mid \alpha, \rho)$ on a 30-point SNR grid (`Bank.prob_of_tk_given_rho`, `_tk_rows_block` and `_tk_kernel_exponent`; matches come from `PNFlatMetric.matches` in `src/sgn_manifold/pn_embedding.py`);
4. the *normalisation and spline fit* (`Bank.mass_model_coefficients`).

Production settings. The O4 p_{astro} build runs the tool on the 1 815 963-template O4 bank with `--window 50000` and `--match-dispersion 0.1`, the default `pn` matches, and the default likelihood kernel with per-signal normalisation. Where a choice below matters, the text says which setting production uses.

Notation. The bank has N templates, sorted by chirp mass. Template j is a rectangle r_j in bank coordinates (for the production coordinate system, $(\log_{10} m_1, \log_{10} m_2, \chi)$) with centre x_j. The coordinate function maps a centre to physical parameters $(m_1, m_2, s_{1z}, s_{2z})$. Throughout, $\kappa_{jk} \in [0, 1]$ denotes the match between templates j and k (the code calls it `k`; the Greek letter keeps it distinct from the template index k), and

$$\chi = \frac{m_1 s_{1z} + m_2 s_{2z}}{m_1 + m_2} \tag{1}$$

is the mass-weighted aligned spin. Masses are ordered so that $m_1 \geq m_2$ (the code swaps components if the centre has $m_2 > m_1$). $\theta_j = (m_1, m_2, \chi)(x_j)$ denotes the intrinsic source parameters at the centre of template j's cell; a source whose parameters fall in that cell is represented by θ_j, so the continuous population is discretised onto the set $\{\theta_j\}$.

Three kinds of object appear and are written differently:

- $P(\cdot)$ is the *probability* of a discrete event: that a source drawn from the population is represented by θ_j, $P(\theta_j \mid \alpha)$; that a signal with parameters θ_j observed at SNR ρ is recovered by template t_k, $P(t_k \mid \theta_j, \rho)$; and their combination $P(t_k \mid \alpha, \rho)$. These are dimensionless numbers that sum to one over j or over k.
- $\pi(\cdot \mid \alpha)$ is a population *probability density* over continuous source parameters, e.g. $\pi(m_1, m_2, \chi \mid \alpha)$, with units of inverse parameter volume. α always denotes the *intrinsic* population, the mergers that occur in the universe; the population a detector sees is derived from it in §5. What the code implements is often only *proportional* to π, with an explicit normalisation step applied where one is needed.

- μ_j is the *unnormalised population mass* of cell j, the integral of π over the cell. This is what the code stores as the per-template "prior"; $P(\theta_j \mid \alpha)$ is μ_j divided by the total mass.

The symbol ρ is reserved for the matched-filter SNR.

2 The discretised population, $P(\theta_j \mid \alpha)$

The population model α describes the intrinsic population: mergers occur at a rate R per unit volume per unit time, with parameters distributed as the probability density

$$\pi(m_1, m_2, \chi \mid \alpha) = \pi(m_1 \mid \alpha)\, \pi(m_2 \mid m_1, \alpha)\, \pi(\chi \mid m_1, m_2, \alpha), \tag{2}$$

whose three factors are given in §2.2 and §2.3. Nothing in Sections 2 to 4 depends on how far the detector can see; the detector's reach enters in §5. The probability that a source drawn from α falls in template j's cell r_j, and is thereby represented by θ_j, is the integral of this density over the cell. The code approximates that integral by the midpoint rule, the density at the cell centre times the physical cell volume V_j:

$$\mu_j \equiv \int_{r_j} \pi(m_1, m_2, \chi \mid \alpha)\, \mathrm{d}m_1\, \mathrm{d}m_2\, \mathrm{d}\chi \;\approx\; \pi\big(m_1(x_j), m_2(x_j), \chi(x_j) \mid \alpha\big)\, V_j. \tag{3}$$

Because the implemented density is only proportional to π, μ_j is an unnormalised mass; the probability that a source is represented by θ_j, which the code calls the template prior, is its share of the total,

$$P(\theta_j \mid \alpha) = \frac{\mu_j}{\sum_{j'=1}^{N} \mu_{j'}}. \tag{4}$$

The code carries the unnormalised μ_j through the recovery model and normalises only at the end (§4). The total mass $\sum_j \mu_j$ is stored as the attribute `log_prior_mass`.

2.1 Cell volume

For the 3-D coordinate system $(\log_{10} m_1, \log_{10} m_2, \chi)$ the physical volume is the product of the mass widths obtained by exponentiating the cell edges, times the χ width:

$$V_j = \big(10^{u_1^{\max}} - 10^{u_1^{\min}}\big)\big(10^{u_2^{\max}} - 10^{u_2^{\min}}\big)\, \Delta\chi, \qquad u_i = \log_{10} m_i. \tag{5}$$

2.2 Mass density

The population is a power law in the primary mass with the secondary uniform between a floor $m_{\min}$ and the primary:

$$\pi(m_1 \mid \alpha) \propto m_1^{\alpha}, \qquad \pi(m_2 \mid m_1, \alpha) = \frac{1}{m_1 - m_{\min}} \quad \text{on } [m_{\min}, m_1]. \tag{6}$$

The code uses m_1^{α} directly, without the constant that would normalise it over the mass range; the conditional on m_2 is a proper density. The missing constant is common to every template and is removed by the normalisation in (4). Here α doubles as the name of the whole population model and the power-law index that parametrises it. The `salpeter` model is the power law with $\alpha = -2.35$; the `power-law` model takes α from `index`. On the equal-mass slices m_2 is pinned, so there is no conditional and the density is simply m^{α}.

The driver requires $m_{\min} < 0.95\, m_2^{\min,\mathrm{bank}}$ and the density asserts $m_2 > 1.05\, m_{\min}$, so the factor $1/(m_1 - m_{\min})$ never approaches a pole.

2.3 Spin density

Component spins are taken independent and uniform, $s_{1z} \sim U[-a, a]$ and $s_{2z} \sim U[-b, b]$, with the half-width switching at the neutron-star mass cut:

$$a = \begin{cases} x_{\mathrm{NS}}^{\max} & m_1 < m_{\mathrm{NS}}^{\max} \\ x_{\mathrm{BH}}^{\max} & \text{otherwise} \end{cases}, \qquad b = \begin{cases} x_{\mathrm{NS}}^{\max} & m_2 < m_{\mathrm{NS}}^{\max} \\ x_{\mathrm{BH}}^{\max} & \text{otherwise} \end{cases}. \tag{7}$$

Defaults are $m_{\mathrm{NS}}^{\max} = 3$, $x_{\mathrm{NS}}^{\max} = 0.05$, $x_{\mathrm{BH}}^{\max} = 1$.

The induced density of $\chi = (m_1 s_{1z} + m_2 s_{2z})/M$ with $M = m_1 + m_2$ is the convolution of two uniforms, a trapezoid. Writing $w_i = m_i/M$ and changing variables from (s_{1z}, s_{2z}) to (χ, s_{2z}) with Jacobian $1/w_1 = M/m_1$,

$$\pi(\chi \mid m_1, m_2, \alpha) = \frac{1}{4ab}\frac{M}{m_1}\int_{-b}^{b} \mathbf{1}\Big[\,\big|\tfrac{M\chi - m_2 s_{2z}}{m_1}\big| \le a\Big]\,\mathrm{d}s_{2z} \tag{8}$$

$$= \frac{M}{4ab\,m_1}\left[\min\Big(\frac{M\chi + m_1 a}{m_2},\ b\Big) - \max\Big(\frac{M\chi - m_1 a}{m_2},\ -b\Big)\right]_+ . \tag{9}$$

The bracket is the length of the s_{2z} interval compatible with the given χ; $[\cdot]_+$ clips negative lengths to zero. This is exactly the expression in `Log10M1Log10M2Chi._spin_density`. Coordinate systems without a χ axis use $\pi(\chi \mid m_1, m_2, \alpha) = 1$.

2.4 Cuts

If a `cuts` block is given, μ_j is set to zero unless the centre satisfies the m_1 and m_2 boxes and the χ box derived from the component-spin cuts by the same weighted mean:

$$\chi_{\min} = \frac{m_1 x_1^{\mathrm{lo}} + m_2 x_2^{\mathrm{lo}}}{M}, \qquad \chi_{\max} = \frac{m_1 x_1^{\mathrm{hi}} + m_2 x_2^{\mathrm{hi}}}{M}. \tag{10}$$

3 The recovery model, $P(t_k \mid \alpha, \rho)$

A signal drawn from the population has intrinsic parameters represented by the centre θ_j of the cell they fall in (up to the optional displacement of (14)). The search recovers it with whichever template has the largest matched-filter SNR. Writing $P(t_k \mid \theta_j, \rho)$ for the probability that a signal with parameters θ_j, observed at SNR ρ, is recovered by template t_k, the recovery model is the discretised population propagated through that kernel:

$$P(t_k \mid \alpha, \rho) = \sum_{j=1}^{N} P(\theta_j \mid \alpha)\, P(t_k \mid \theta_j, \rho). \tag{11}$$

Since $\sum_k P(t_k \mid \theta_j, \rho) = 1$ for every j (§3.2), the right-hand side sums to one over k automatically; the code works with the unnormalised μ_j in place of $P(\theta_j \mid \alpha)$ and divides out the total mass at the end.

3.1 The likelihood-ratio kernel

Let the noise-free data be a signal with parameters θ_j at SNR ρ, and let $\hat{h}_k$ be the unit-norm template k. The log likelihood ratio for template k at amplitude A is $A\langle d \mid \hat{h}_k\rangle - A^2/2$; maximising over A gives $\rho_k^2/2$, where $\rho_k = \langle d \mid \hat{h}_k\rangle$ is the SNR observed at template k. The signal's waveform is template j's, so the noise-free expectation of that SNR is $\rho_k = \rho\,\kappa_{jk}$, so

$$\ln \Lambda_k = \tfrac{1}{2}\rho^2 \kappa_{jk}^2. \tag{12}$$

The recovery kernel is this likelihood ratio divided by its value at the true template, $e^{\rho^2/2}$. It is written as e^{-g} with an exponent proportional to a mismatch argument D:

$$w_{jk}(\rho) = \exp\Big[-\tfrac{\rho^2}{2}\,(1-\kappa_{jk}^2)\Big] = e^{-g(\rho,\kappa_{jk})}, \qquad g = \tfrac{\rho^2}{2}\,D_{jk}, \qquad D_{jk} = 1-\kappa_{jk}^2. \tag{13}$$

D is 0 at a perfect match and 1 at zero match, and it is first order in mismatch near $\kappa = 1$ ($1-\kappa^2 \approx 2(1-\kappa)$). The historical kernel behind `--legacy-kernel` is described in Appendix A.

Real signals do not sit exactly on a template: the bank is discrete and the waveform manifold is not fully covered by the approximant. The option `--match-dispersion` ϵ (default 0) lets the signal sit off template j. For each template k its mismatch is treated as $D' = (\sqrt{D} + \sigma\xi)^2$ with ξ standard normal and $\sigma^2 = \epsilon\,(1-D)$, and the kernel weight $e^{-\rho^2 D/2}$ is replaced by its expectation over ξ. The offset is drawn independently for each pair and only along the direction from j to k, so this is a pairwise, one-dimensional blurring rather than a marginalisation over a displaced signal. The exponent becomes

$$g_{\text{eff}}(\rho, D) = \frac{\rho^2}{2}\,\frac{D}{1+t} + \frac{1}{2}\ln(1+t), \qquad t = \rho^2\epsilon\,(1-D). \tag{14}$$

At $\epsilon = 0$ this is the exponent of (13). At $D = 1$ it is $\rho^2/2$ for every ϵ, so unrelated templates keep full suppression. For small D it saturates at $D/(2\epsilon)$ as ρ grows, so templates within about ϵ mismatch of the signal stay indistinguishable at any SNR. The derivation is in Appendix B.

3.2 Per-signal normalisation

$P(t_k \mid \theta_j, \rho)$ is a distribution over the recovered template for a signal with fixed parameters θ_j, so the kernel is normalised over k:

$$P(t_k \mid \theta_j, \rho) = \frac{w_{jk}(\rho)}{Z_j(\rho)}, \qquad Z_j(\rho) = \sum_{k=1}^{N} w_{jk}(\rho). \tag{15}$$

Z_j is indexed by the signal's parameters θ_j, not by the recovered template. It is the kernel-weighted count of templates within reach of j, and it enters the recovered-template distribution only through the mixture (11), where each signal's contribution to template k is divided by that signal's own Z_j.

Previous versions of the code omitted Z_j, so the recovery probabilities of a signal did not sum to one and the model was distorted in a template-dependent way. Appendix A.2 works out what the omission does and quantifies it on a production-style bank.

The code computes, for each ρ on the grid and each recovered template k,

$$\log P(t_k \mid \alpha, \rho) = \operatorname{logsumexp}_j\Big[\log \mu_j - \log Z_j(\rho) - g(\rho, \kappa_{jk})\Big] + \text{const}. \tag{16}$$

The constant does not depend on k and is removed by the normalisation over k in (27), so it is ignored.

How Z_j is evaluated, and the one place it is approximated, is described at the end of §3.4, since both depend on the chirp-mass window.

3.3 The match model

The default `--match-method pn` scores template pairs with the flat metric in TaylorF2 phase-coefficient space. Each template is mapped to a coefficient vector c on the basis $\{f^{(n-5)/3},\ n = 0..7,\ n \neq 5\} \cup \{\ln f,\ f^{1/3}\ln f\}$, and

$$\kappa_{jk} = \min\big(1,\ \exp[-\delta c^{\mathsf{T}} G\,\delta c]\big), \qquad \delta c = c_k - c_j, \tag{17}$$

where G is one half of the moment matrix of the basis functions under the whitened-amplitude weight $f^{-7/3}/S_n(f)$, with the subspace $\mathrm{span}\{1, f\}$ projected out; that projection is the analytic maximisation over t_c and ϕ_c. G is tabulated on a grid of lower band edges, and each pair is scored on the band that starts at the larger of its two lower cutoffs, $\max(f_{\mathrm{low},j}, f_{\mathrm{low},k})$. Equal-match surfaces are globally quadratic in δc.

The construction is the flat metric in the space of post-Newtonian phase coefficients of Brown et al. [4], which applies the t_c, ϕ_c projection of the Owen metric [1, 2] in coefficient space; coordinates in which the inspiral metric is constant go back to Tanaka and Tagoshi [3]. Two details are specific to this code: the exponential form of (17), which extends the quadratic mismatch beyond its small-mismatch range, and the per-pair lower band edge.

3.4 The chirp-mass window and the far-field term

Evaluating (16) exactly costs one match per template pair, N^2 in all, which is out of reach for a production bank ($N = 1.8$M templates, 3×10^{12} pairs). The window exploits the fact that the match falls to zero as two templates separate in chirp mass. With the bank sorted by chirp mass, let W_k be the n templates nearest to k in that ordering (`--window` n). For j outside W_k the code takes $\kappa_{jk} = 0$, so $D_{jk} = 1$ and $w_{jk} = e^{-\rho^2/2}$, the same weight for every far template. Splitting the sum in (11), written with the unnormalised masses, at the window boundary:

$$\sum_{j=1}^{N} \frac{\mu_j\, w_{jk}}{Z_j} = \sum_{j \in W_k} \frac{\mu_j\, e^{-g(\rho,\kappa_{jk})}}{Z_j} \;+\; e^{-\rho^2/2} \sum_{j \notin W_k} \frac{\mu_j}{Z_j}. \tag{18}$$

The first sum is computed with explicit matches. The second needs no matches at all: it is the prior mass outside the window, available exactly from prefix sums of μ_j over the chirp-mass ordering, except that the per-signal Z_j cannot be carried through a prefix sum. It is replaced by a single typical value $\bar{Z}(\rho)$, defined in the paragraph on computing Z_j below, giving what the code evaluates:

$$P(t_k \mid \alpha, \rho) \propto \sum_{j \in W_k} \frac{\mu_j}{Z_j(\rho)}\, e^{-g(\rho,\kappa_{jk})} \;+\; \frac{e^{-\rho^2/2}}{\bar{Z}(\rho)} \sum_{j \notin W_k} \mu_j. \tag{19}$$

Two approximations enter, and both vanish when the window covers the bank: that $\kappa_{jk} = 0$ for every j outside W_k, which requires W_k to contain the whole match > 0 neighbourhood of k; and $Z_j \to \bar{Z}$ for the far templates. The far-field term itself matters only at low ρ. At $\rho = 10$ each far template carries e^{-50}, and even 1.8M of them add up to nothing against the near sum, so the truncated computation is exact there; at $\rho = 3.16$ each carries e^{-5} and the far field is the bulk of the sum.

Production uses $n = 50\,000$ on the 1.8M-template O4 bank, so the far field holds 97% of the templates. The same window on the 1.04M-template all-sky bank was validated against the full computation: within 0.03 in $\log P$ at low ρ and exact at $\rho \geq 10$. The default $n = 2\,000\,000$ covers any current bank and makes (19) exact.

Computing Z_j. The normalisation (15) is a sum of the same kernel over k, so it is evaluated by the same row machinery with the roles of the indices exchanged: the row is now the signal's cell j, the window W_j is centred on j, and every template's prior is replaced by one. The window is truncated in the same way as (19). Templates outside W_j have $\kappa_{jk} \approx 0$ and contribute $e^{-\rho^2/2}$ each, so what the code stores is

$$Z_j(\rho) = \sum_{k \in W_j} e^{-g(\rho,\kappa_{jk})} \;+\; \big(N - |W_j|\big)\, e^{-\rho^2/2}, \tag{20}$$

on the same 30-point ρ grid, with the same `pn` matches as the main pass. Those matches are symmetric, so the kernel summed here is the same one that appears in the numerator of (15). With the default full-bank window the second term vanishes and Z_j is exact.

Z_j is not itself normalised. The self term $w_{jj} = 1$ makes $Z_j \geq 1$, with $Z_j \to 1$ as $\rho \to \infty$ and $Z_j \to N$ as $\rho \to 0$. What is normalised is each signal's recovery distribution: with the full-bank window,

$$\sum_{k=1}^{N} P(t_k \mid \theta_j, \rho) = 1 \qquad \text{for every } j \text{ and } \rho, \tag{21}$$

exactly, up to the single precision in which $\log Z_j$ is stored (a relative error of order 10^{-7}).

Which rows are needed. A job that processes recovered templates $k \in [k_0, k_1)$ needs Z_j for every signal θ_j whose cell falls in some window W_k, that is for $j \in [k_0 - n/2,\ k_1 + n/2) \cap [0, N)$. The Z pass computes exactly that range before the main pass and stores it in single precision. A partial job is therefore self-contained, at the cost of roughly doubling its work.

The one approximation. The far-field term of (19) sums μ_j over $j \notin W_k$ through prefix sums, which cannot carry a different Z_j for each j. Every far template is therefore divided by the same typical value, the median of the computed $Z_j(\rho)$ at that ρ:

$$\bar{Z}(\rho) = \mathrm{median}_{j \in [k_0 - n/2,\ k_1 + n/2)}\, Z_j(\rho). \tag{22}$$

Let n_j be the number of recovered templates k whose window W_k contains j. The recovery distribution actually implemented for a signal with parameters θ_j is then w_{jk}/Z_j for those n_j templates and $e^{-\rho^2/2}/\bar{Z}$ for the remaining $N - n_j$, so its total is

$$\sum_{k=1}^{N} P(t_k \mid \theta_j, \rho) = \frac{1}{Z_j} \sum_{k:\, j \in W_k} w_{jk} + \left(N - n_j\right) \frac{e^{-\rho^2/2}}{\bar{Z}}. \tag{23}$$

Away from the bank's edges the windows are symmetric, so the templates whose windows contain j are exactly the members of W_j and $n_j = |W_j|$; substituting (20) then gives

$$\sum_{k=1}^{N} P(t_k \mid \theta_j, \rho) = 1 - \left(N - |W_j|\right) e^{-\rho^2/2} \left(\frac{1}{Z_j} - \frac{1}{\bar{Z}}\right). \tag{24}$$

This is exactly one when the window covers the bank or when $Z_j = \bar{Z}$. Otherwise the deviation is the far-field share of the normalisation times the relative difference between Z_j and $\bar{Z}$. It vanishes at high ρ, where $e^{-\rho^2/2}$ is negligible, and is largest at low ρ for signals θ_j near a bank edge, where up to half the kernel falls outside the bank and Z_j can be about half of $\bar{Z}$. Equations (21), (23) and (24) were checked against the row kernel on a synthetic bank. On the 1.04M-template production bank the code's validation with a 50 000 window found $\log P(t_k \mid \alpha, \rho)$ within 0.03 of the full computation at low ρ, and its docstring records Z_j as constant to about 1% in that regime; the default full-bank window avoids the approximation altogether.

3.5 Behaviour across the SNR grid

The grid is $\rho_i = 10^{0.5 + i/29}$, $i = 0..29$, i.e. 30 log-spaced values from 3.16 to 31.6. The two limits of the normalised model are instructive:

$$\rho \to 0: \quad w_{jk} \to 1, \qquad Z_j \to N, \qquad P(t_k \mid \alpha, \rho) \to \frac{1}{N}, \tag{25}$$

$$\rho \to \infty: \quad w_{jk} \to \delta_{jk}, \qquad Z_j \to 1, \qquad P(t_k \mid \alpha, \rho) \to P(\theta_k \mid \alpha). \tag{26}$$

In the first limit recovery is random and the prior is forgotten; in the second recovery is exact and the prior is returned unchanged. At the low end of the grid $\rho^2/2 \approx 5$, so a neighbour at match 0.9 keeps about 40% of the weight and $P(t_k)$ is the prior blurred over a wide neighbourhood. At the high end $\rho^2/2 = 500$ and a neighbour at match 0.99 is suppressed by e^{-10}.

4 Normalisation and the spline

For each ρ_i the row is shifted by its log-sum-exp over templates,

$$\log P(t_k \mid \alpha, \rho_i) \;\leftarrow\; \log P(t_k \mid \alpha, \rho_i) - \operatorname{logsumexp}_{k'} \log P(t_{k'} \mid \alpha, \rho_i), \tag{27}$$

so that $\sum_k P(t_k \mid \alpha, \rho_i) = 1$. This is where the unnormalised cell masses μ_j of §2 and the k-independent constant of (16) drop out. Then, for each template k, the 30 values of $\log P(t_k \mid \alpha, \rho_i)$ are fit with a cubic B-spline in ρ using SciPy's `splrep`, then converted to piecewise cubic polynomials with `PPoly.from_spline`. The stored arrays are

$$\texttt{coefficients}[4,\ n_\rho + 3,\ N], \qquad \texttt{SNR}[n_\rho + 4]\ \text{(breakpoints)}. \tag{28}$$

Any $-\infty$ entries (templates with zero prior and no contributing neighbours) are replaced by the smallest finite value minus $\ln 10^6$ before fitting so the spline stays finite. The ranking statistic evaluates this spline at the observed SNR to obtain the signal hypothesis's probability of landing on template k. Coefficients are fit only when the job covers the full bank (`start-ix` $= 0$, `stop-ix` $= N$); partial jobs store the raw grid for a later merge.

5 Selection effects: from the merger rate to the detected rate

Everything up to this point describes the intrinsic population: α is the distribution of mergers in the universe, and the stored recovery model is built from it. A detector does not see that population. A source is detected only if it lies within the detector's reach, and that reach depends on the source parameters, so the detected population is the intrinsic one reweighted by the volume in which each kind of source can be seen. This section gives that reweighting and explains why the model file stores it separately from the recovery model.

5.1 The sensitive volume and the detected population

Let $D(\theta)$ be the horizon distance of a source with parameters $\theta = (m_1, m_2, \chi)$: the distance at which an optimally located and oriented source produces SNR 8 in the bank's PSD. Ignoring cosmology, the volume within which such a source is detectable is

$$V(\theta) = \tfrac{4\pi}{3}\, D(\theta)^3, \tag{29}$$

up to a factor from averaging over sky location and orientation that is the same for every θ and cancels in every ratio below. If mergers occur at a rate R per unit volume per unit time with parameters distributed as $\pi(\theta \mid \alpha)$, the rate of detected mergers with parameters θ is

$$\Lambda(\theta) = R\, \pi(\theta \mid \alpha)\, V(\theta), \tag{30}$$

and the population of detected sources is

$$\pi_{\mathrm{det}}(\theta \mid \alpha) \propto \pi(\theta \mid \alpha)\, D(\theta)^3. \tag{31}$$

A heavy binary is intrinsically rarer under a falling mass power law, but its horizon grows with mass through the inspiral regime, roughly as $\mathcal{M}_c^{5/6}$ at fixed PSD, so its sensitive volume grows roughly as $\mathcal{M}_c^{5/2}$ and heavy binaries are over-represented among detections.

In template terms, with D_j the horizon of template j and D_{can} the horizon of a $1.4+1.4\,M_\odot$ non-spinning binary on the same PSD,

$$\mu_j^{\mathrm{det}} = \mu_j \left(\frac{D_j}{D_{\mathrm{can}}}\right)^3, \qquad M = \sum_j \mu_j, \qquad M_{\mathrm{det}} = \sum_j \mu_j^{\mathrm{det}}, \tag{32}$$

Like μ_j, $\mu_j^{\rm det}$ is an unnormalised mass: it discretises $\pi_{\rm det}$ of (31) onto the template grid exactly as μ_j discretises π in (3), since D is evaluated at the cell centre. The detected population on the grid is its normalised form, the counterpart of (4),

$$P_{\rm det}(\theta_j \mid \alpha) = \frac{\mu_j^{\rm det}}{M_{\rm det}} = \frac{P(\theta_j \mid \alpha)\,(D_j/D_{\rm can})^3}{\sum_{j'} P(\theta_{j'} \mid \alpha)\,(D_{j'}/D_{\rm can})^3}. \tag{33}$$

Integrating (30) over θ gives the total detected rate. Factoring $V(\theta) = V_{\rm can}(D/D_{\rm can})^3$ with $V_{\rm can} = \frac{4\pi}{3} D_{\rm can}^3$, the remaining integral is the population average of $(D/D_{\rm can})^3$, which on the grid is a ratio of the two masses:

$$\Lambda = R\,V_{\rm can}\left\langle (D/D_{\rm can})^3 \right\rangle_\alpha, \qquad \left\langle (D/D_{\rm can})^3 \right\rangle_\alpha = \sum_j P(\theta_j \mid \alpha) \left(\frac{D_j}{D_{\rm can}}\right)^3 = \frac{M_{\rm det}}{M}. \tag{34}$$

The two scalars the file stores are $\ln M$ (`log_prior_mass`) and $\ln M_{\rm det}$ (`log_det_prior_mass`); their difference is the log of the population-averaged sensitive volume in units of the canonical BNS volume, and (33) is recovered from the stored μ_j and D_j.

Table 1 shows the reweighting on the small-sky Salpeter example of §A.4, from the stored μ_j and D_j. Two thirds of the intrinsic mergers are in the BNS bin and under one percent of the detections are; the population-averaged volume is 61 canonical-BNS volumes.

$\mathcal{M}_c$ bin	intrinsic fraction	median $D/D_{\rm can}$	detected fraction
$[0.8, 1.5)$	0.650	0.95	0.009
$[1.5, 3)$	0.228	1.53	0.017
$[3, 6)$	0.075	2.40	0.024
$[6, 12)$	0.028	3.89	0.044
$[12, 25)$	0.013	6.05	0.093
$[25, 50)$	0.004	9.41	0.151
$[50, 200)$	0.002	18.2	0.663

Table 1: Intrinsic population mass μ_j/M and detected mass $\mu_j^{\rm det}/M_{\rm det}$ summed over chirp-mass bins for the small-sky Salpeter model, with $D_{\rm can} = 213.7\,$Mpc.

5.2 Why the stored model is intrinsic

The recovery model of §3 is conditioned on the SNR ρ, and conditioning on ρ already selects the detected population. For a population uniform in Euclidean volume, the number of sources with SNR in $[\rho, \rho + {\rm d}\rho]$ scales as $D(\theta)^3\rho^{-4}\,{\rm d}\rho$, and the ρ^{-4} is the same for every θ, so

$$p(\theta \mid \rho) \propto \pi(\theta \mid \alpha)\,D(\theta)^3 \qquad \text{independently of } \rho. \tag{35}$$

The population appropriate to a signal known to have SNR ρ is therefore $P_{\rm det}(\theta_j \mid \alpha)$ of (33), not $P(\theta_j \mid \alpha)$. The stored model nevertheless uses μ_j, for two reasons.

SGNL adds the selection effect on the fly. The first consumer of the file is SGNL's ranking statistic, which applies the selection factor itself as a per-template D_H^3 weight on the recovered template, using its own horizons. A mass model that already carried the factor would count it twice. The mass model therefore stores only intrinsic-population terms.

$p_{\rm astro}$ **needs detected-signal weights.** The second consumer is the $p_{\rm astro}$ calculation in this repository, and its inputs refer to detected signals: the detected rate of each source category through $\langle (D/D_{\rm can})^3 \rangle_a$ in (34), the split of a total rate among categories through their intrinsic masses M_a, and the detected recovery model of §5.3. Those cannot be formed from the intrinsic model alone. That is why the file carries D_j, $D_{\rm can}$, M and $M_{\rm det}$ beside the recovery model: the intrinsic pieces serve SGNL unchanged, and the extra pieces let $p_{\rm astro}$ form the detected quantities at read time.

5.3 The detected recovery model

There are two ways to attach the selection factor to the recovery model. Weighting the *signal* side before smearing follows directly from (35): it is (11) with the detected population (33) in place of the intrinsic one,

$$P_{\mathrm{det}}(t_k \mid \alpha, \rho) = \sum_{j=1}^{N} P_{\mathrm{det}}(\theta_j \mid \alpha)\, P(t_k \mid \theta_j, \rho). \tag{36}$$

Weighting the *recovered* template after smearing is SGNL's runtime convention, and p_{astro} follows it so that the factor cancels between a category and the population:

$$P_{\mathrm{det}}(t_k \mid \alpha, \rho) = \frac{(D_k/D_{\mathrm{can}})^3\, P(t_k \mid \alpha, \rho)}{N(\rho)}, \qquad N(\rho) = \sum_k (D_k/D_{\mathrm{can}})^3\, P(t_k \mid \alpha, \rho). \tag{37}$$

The two agree when D is constant across the width of the kernel, so at high ρ, and differ at low ρ, where the kernel mixes templates of different horizon. Table 2 measures the difference on the small-sky example of §A.4, using the stored D_j and the same row kernel (`det\_weighting.py` alongside these notes). For $\rho \geq 8$ the two agree per template to within a few percent. Below that they diverge quickly: at $\rho = 3.16$ the intrinsic model is nearly uniform over templates, as the $\rho \to 0$ limit of §3.5 requires, so (36) is nearly uniform too, while (37) is proportional to D_k^3 and puts half its probability in the heaviest chirp-mass bin. p_{astro} evaluates the recovery model at an SNR inferred from the false-alarm rate, which is never below about 7.6, so it operates where the two agree.

ρ	p_1	p_{50}	p_{99}	total variation
3.16	0.015	0.13	9.2	0.76
5	0.018	0.14	2.2	0.17
8	0.98	1.03	1.05	0.066
10	0.98	1.02	1.04	0.050
15	0.98	1.01	1.02	0.028
31.6	0.99	1.00	1.01	0.009

Table 2: Ratio of the recovered-weighted model (37) to the signal-weighted model (36) per template (percentiles over templates with $\mu_k > 0$), and the total variation distance $\frac{1}{2}\sum_k |P_{\mathrm{after}} - P_{\mathrm{before}}|$ between the two distributions, on the small-sky Salpeter model.

5.4 Computing D_j and D_{can}

An exact horizon distance for every template would need a full-resolution waveform and a noise-weighted integral for each of the 1.8M templates. The code instead takes the *shape* of D_j across the bank from a cheap proxy and fixes its *scale* with three exact evaluations.

The proxy. The low-resolution waveform store that refinement already uses generates each template on its coarse frequency grid at a reference distance of 1 Mpc and whitens it by the PSD; σ_j is the norm of that whitened waveform. The optimal SNR of the same source at distance r is $\sigma_j \times (1\,\mathrm{Mpc}/r)$ up to a constant set by the grid's conventions, so the horizon at SNR 8 is proportional to σ_j with the same unknown constant for every template.

The scale. `sgnligo.psd.HorizonDistance` computes the horizon exactly, on a fine frequency grid with the correct normalisation, for one set of source parameters. The code runs it for three probe templates chosen by sorting the bank by σ and taking the templates at the 10th, 50th and 90th percentiles, so that the probes span the bank's range of sensitivity. Each

probe gives a ratio of exact horizon to proxy; if the proxy were exact up to a constant the three ratios would agree. Their median is the calibration constant, applied to every template, and their spread is stored as a check on the proxy:

$$r_p = \frac{D_p^{\text{fine}}}{\sigma_p}, \qquad \text{cal} = \text{median}_p\, r_p, \qquad D_j = \text{cal}\,\sigma_j, \qquad \text{spread} = \frac{\max_p r_p - \min_p r_p}{\text{cal}}, \tag{38}$$

all at SNR 8. Every grid and whitening constant cancels in r_p, so the spread measures only the template-dependent error of the low-resolution proxy; it is 0.2% on the small-sky bank.

The canonical horizon. D_{can} is the same exact evaluation for a $1.4 + 1.4\,M_\odot$, $\chi = 0$ binary at SNR 8 on the same PSD and configuration. It is not tied to any template, so it does not depend on the bank's coverage.

5.5 Assumptions

- Volumes are Euclidean and no redshift is applied; this matters most for the heaviest binaries, which are seen at cosmological distances.
- D is the horizon at optimal location and orientation. The angular-averaging factor is common to all templates and cancels in (31)–(34).
- The PSD shape is frozen at bank build. p_{astro} supplies the live scale through a single range input, so only the relative sensitivity across the bank is taken from the file.
- In (37) the factor is evaluated at the recovered template rather than at the signal's true parameters.

6 Output file layout

Name	Kind	Content
`valid_templates`	dataset	indices [`start-ix`, `stop-ix`) processed by this job
`rhos`	dataset	the 30-point SNR grid
`log_p_of_tk_given_alpha_rho`	dataset	$(30, N)$ raw $\log P(t_k \mid \alpha, \rho)$; NaN outside the job's slice
`template_id`	dataset	bank ids in bank (chirp-mass) order
`m1s`, `m2s`, `chis`	dataset	physical centre of each template, $m_1 \geq m_2$
`priors`	dataset	cell masses μ_j from (3), unnormalised
`horizons_mpc`	dataset	D_j at SNR 8 (float32)
`SNR`	dataset	spline breakpoints (full-bank jobs only)
`coefficients`	dataset	$(4, n_\rho + 3, N)$ piecewise-cubic coefficients of $\log P$ (full-bank jobs only)
`mass_model_spec`	attr	JSON of the model block (provenance)
`canonical_bns_horizon_mpc`	attr	D_{can}
`horizon_snr`	attr	8.0
`horizon_calibration_spread`	attr	probe spread
`log_prior_mass`	attr	$\ln \sum_j \mu_j$
`log_det_prior_mass`	attr	$\ln \sum_j \mu_j (D_j / D_{\text{can}})^3$

7 Remarks and points for discussion

Remark 1 (Midpoint rule)**.** *Equation* (3) *evaluates the density at the cell centre rather than integrating it over the cell. For a steep power law and wide high-mass cells the two differ; the effect is largest where cells are widest in* m_1.

Remark 2 (Zero-prior templates)**.** *A template with $\mu_k = 0$ still receives probability from its neighbours through (11); it is not excluded from recovery. Only the row-wise $-\infty$ guard in §4 treats templates that receive nothing at all.*

Remark 3 (Noise in recovery)**.** *The kernel of §3.1 is evaluated at the expected SNRs $\rho\,\kappa_{jk}$ and neglects the $\mathcal{O}(1)$ noise fluctuations of the realised SNRs; it is not the probability that template k has the largest noisy SNR. At high ρ those fluctuations are small relative to the $\rho^2(1-\kappa^2)/2$ contrast between neighbouring templates, but at the low end of the grid ($\rho \approx 3$) they are comparable to it, and the true recovery distribution is broader than the model's. Whether this matters for the ranking statistic is worth discussing.*

Remark 4 (Spline range)**.** *The spline is fit on $[3.16, 31.6]$. Evaluations outside that range extrapolate the end polynomials; the consumer should clamp or be aware.*

A The legacy model

The flag `--legacy-kernel` reproduces the historical model exactly. It differs from the production model of §3.1 and §3.2 in two ways: a different kernel, and no per-signal normalisation.

A.1 The squared kernel

The kernel is a Gaussian in the distance $\rho\,(1-\kappa_{jk})$ rather than a likelihood ratio:

$$w_{jk}^{\text{leg}}(\rho) = \exp\!\left[-\tfrac{\rho^2}{2}\,(1-\kappa_{jk})^2\right], \qquad D_{jk}^{\text{leg}} = (1-\kappa_{jk})^2. \tag{39}$$

It has no likelihood interpretation. Near $\kappa = 1$ it is second order in mismatch where the production kernel is first order, so at a given ρ it suppresses close neighbours far less. The two kernels share the far-field limit $g(\rho, 0) = \rho^2/2$, so the window and far-field machinery of §3.4 applies unchanged, and the match-dispersion form (14) is applied to D^{leg} in the same way (`_tk_kernel_exponent` handles both).

A.2 Omitting the normalisation

The historical model set $Z_j \equiv 1$ in (15), so that

$$P^{\text{leg}}(t_k \mid \alpha, \rho) \propto \sum_j \mu_j\, w_{jk}(\rho), \tag{40}$$

and a signal with parameters θ_j is recovered with total probability $\sum_k w_{jk} = Z_j$ rather than one. The final normalisation over k (§4) is still applied, so (40) is a proper distribution at each ρ, but it removes only a constant and any k-dependent distortion survives it.

To see the distortion, work in the regime where the kernel is wider than the template spacing but narrower than the scale on which the population and the bank's coverage vary. Write the mismatch between nearby points as $\mathrm{d}s^2 = \delta x^{\mathsf{T}} g\, \delta x$, so that $w \approx e^{-\rho^2 \mathrm{d}s^2}$, and let $\mathcal{V}_\rho = \int e^{-\rho^2 \mathrm{d}s^2}\, \mathrm{d}^d s \propto \rho^{-d}$ be the kernel's mismatch volume. Let $n(x)$ be the number of templates per unit mismatch volume, so that a cell's physical volume is $V_j \approx 1/(n_j \sqrt{\det g_j})$ and $\mu_j \approx \pi(x_j)\, V_j$. The sum over j against the cell volumes is a Riemann sum,

$$\sum_j \mu_j\, w_{jk} \approx \int \pi(x)\, w(x, x_k)\, \mathrm{d}x \approx \frac{\pi(x_k)}{\sqrt{\det g_k}}\, \mathcal{V}_\rho, \tag{41}$$

while the normalisation in the same regime is $Z_j \approx n_j\, \mathcal{V}_\rho$. Since Z varies slowly across the kernel, the production model gives

$$\sum_j \frac{\mu_j}{Z_j}\, w_{jk} \approx \frac{\pi(x_k)}{n_k \sqrt{\det g_k}} \approx \mu_k, \tag{42}$$

the cell mass smoothed over the kernel, whereas the legacy model gives $\mu_k Z_k$: the cell mass multiplied by the number of templates within the kernel's reach of k.

Two consequences follow. First, Z measures template density in match space, not in physical coordinates. A bank placed at uniform minimal match has many more templates per unit mass at low mass than at high mass, but the kernel is correspondingly narrower there in physical coordinates, and the number of templates inside it is the same. The physical density gradient across the mass range therefore does not by itself tilt the legacy model toward either end of the population. Second, what does tilt it is non-uniform coverage in match space. Regions that are over-covered in true match, for example where refinement added templates or where the placement metric over-estimated distances, have a larger Z and are inflated; regions near the bank's edges, where part of the kernel falls outside the bank and Z is smaller, are deflated. At high ρ the kernel is narrower than the spacing and $Z \to 1$, so the two models agree, unless the bank contains near-duplicate templates; refinement produces them, and then the legacy model double counts each duplicated pair at every SNR. The next section measures all of this on a real bank.

A.3 The local-metric match option

`--match-method metric` scores pairs with the local placement metric g_k at the recovered template, $\kappa_{jk} = \exp[-\delta x^{\mathsf{T}} g_k\, \delta x]$ with $\delta x = x_j - x_k$, instead of (17). It is kept for A/B comparison only: a local quadratic form in chart coordinates mis-measures matches along the q–χ degeneracy valley, which is flat in coefficient space. It is also not symmetric. The main pass uses g_k while the Z pass, whose rows are indexed by j, uses g_j, so w_{jk} in the numerator of (15) and in Z_j are not the same function. Neither issue arises in the `pn` path.

A.4 Measurement on a production-style bank

Setup. The bank is `examples/small-sky/small-sky-bank-refined.h5`: 213 007 templates in $(\log_{10} m_1, \log_{10} m_2, \chi)$ with $m_{1,2} \in [1, 200]$, $q \leq 8$, $\chi \in [-0.15, 0.15]$, minimal match 0.97, IMRPhenomD. The prior is the Salpeter model of the example mass model (`examples/small-sky/mass-model.h5`), under which 179 276 templates have $\mu_j > 0$. Z_j and the recovered-template model with and without Z were computed with the production row kernel (`pn` matches, likelihood kernel, $\epsilon = 0$; production uses $\epsilon = 0.1$, which blurs templates within 0.1 mismatch together and so changes Z_j at high ρ, where the near-duplicates would no longer stand alone) on a window of 50 000 templates, as in production, which is exact at $\rho \geq 10$; at lower ρ the far-field count in (20) is exact and only the match ≈ 0 assumption enters. The scripts `z\_survey.py` and `z\_analyze.py` alongside these notes reproduce every number below (about 20 minutes on 6 processes). The three named templates are the ones nearest to $(1.4, 1.4)$, $(10, 1.4)$ and $(30, 30)\, M_\odot$ at $\chi \approx 0$. One caveat: this bank uses IMRPhenomD but is scored, as in production, with TaylorF2 phase coefficients, so matches between high-mass templates where merger and ringdown enter the band are approximate and the top chirp-mass bin below is indicative only.

How much Z_j varies. Table 3 gives percentiles of Z_j over the bank and its value at the three templates. Three regimes are visible. At the bottom of the grid Z_j is dominated by the far field, $Ne^{-\rho^2/2} \approx 1435$, and is the same for every template to 1%; this is why the median $\bar{Z}$ is safe there, and it also means the recovered-template distribution is nearly uniform at $\rho = 3.16$, as the $\rho \to 0$ limit of §3.5 says. At $\rho = 10$ the kernel spans a few templates, and Z_j varies by a factor of 12.6 across the bank, with 98% of templates between 1.02 and 4.57. The 1.4+1.4 template sits 41% above the median while the other two are within 6% of it. At the top of the grid the bulk has $Z_j = 1$, but 20% of templates still have $Z_j > 1.1$ and 10% have a neighbour

ρ	p_1	p_{50}	p_{99}	max/min	1.4+1.4	10+1.4	30+30
3.16	1452	1460	1466	1.03	1461	1458	1459
5	3.12	6.10	10.4	14.1	7.17	5.91	6.81
8	1.14	2.46	5.56	14.7	3.28	2.64	2.80
10	1.02	1.87	4.57	12.6	2.64	1.98	1.89
15	1.00	1.26	3.46	9.4	2.04	1.32	1.11
31.6	1.00	1.00	2.50	7.7	1.74	1.00	1.00

Table 3: $Z_j(\rho)$ on the refined small-sky bank: percentiles over all templates, the ratio of the largest to the smallest value, and the value at three templates.

at match above 0.999 (i.e. $Z_j - 1 > e^{-1}$ at $\rho = 31.6$). These are near-duplicates produced by refinement, and the 1.4+1.4 template is one of them.

The distortion per template. Table 4 gives the ratio of the recovered-template probability without Z to the probability with it, both normalised over k at each ρ. Because both distribu-

ρ	p_1	p_{50}	p_{99}	1.4+1.4	10+1.4	30+30
3.16	1.00	1.00	1.00	1.00	1.00	1.00
5	0.69	0.90	1.34	1.06	0.89	0.93
8	0.42	0.82	1.71	1.08	1.10	0.89
10	0.44	0.80	1.83	1.10	1.22	0.80
15	0.59	0.76	1.94	1.19	1.08	0.66
31.6	0.80	0.80	1.89	1.40	0.81	0.80

Table 4: Ratio of the recovered-template probability without Z to the probability with it, $P^{\text{no } Z}(t_k \mid \alpha, \rho)/P(t_k \mid \alpha, \rho)$, both normalised over templates: percentiles over templates with $\mu_k > 0$, and the value at three templates.

tions sum to one, the P-weighted mean of the ratio is exactly one; a median below one means the bulk of templates is deflated to feed a minority of over-covered ones. At $\rho = 10$ the central 98% of templates spans 0.44 to 1.83. At $\rho = 31.6$ the bulk sits at 0.80 and the near-duplicate templates at up to 1.9: the legacy normalisation gives a duplicated pair roughly twice its share at every SNR, so the two models do not converge at the top of the grid on a refined bank. The log ratio correlates with $\log Z_k$ at 0.93 at $\rho = 10$; the residual (median $|\Delta \log| = 0.24$) is the neighbours' Z_j and μ_j entering the mixture.

The tilt across the mass range. Table 5 sums the recovered probability over chirp-mass bins at $\rho = 10$. Two things stand out. With Z, the recovered probability per bin equals the prior mass per bin to three decimals: the normalised model conserves population mass at the bin level, as the continuum argument above says it must. Without Z, the bins at chirp mass 12 to 50 $M_\odot$ lose 25 to 29% of their probability, the 1.5 to 3 $M_\odot$ bin gains 17%, the BNS bin loses 6%, and the top bin, holding 0.2% of the mass, nearly doubles. The median Z_j per bin falls from 2.0 below $\mathcal{M}_c = 1.5$ to 1.4 at 25 to 50, so the match-space coverage of this refined bank is mildly denser at low mass, and the legacy normalisation did tilt the population toward low chirp mass. The tilt tracks Z, that is the coverage in match space, as the argument above predicts.

$\mathcal{M}_c$ bin	templates	prior mass	with Z	without Z	ratio
$[0.8, 1.5)$	32 083	0.650	0.650	0.610	0.94
$[1.5, 3)$	95 618	0.228	0.228	0.265	1.17
$[3, 6)$	37 813	0.075	0.075	0.077	1.02
$[6, 12)$	9 989	0.028	0.028	0.031	1.11
$[12, 25)$	2 718	0.013	0.013	0.010	0.75
$[25, 50)$	772	0.004	0.004	0.003	0.71
$[50, 200)$	283	0.002	0.002	0.004	1.85

Table 5: Total recovered-template probability per chirp-mass bin at $\rho = 10$, with the fraction of the prior mass in each bin for comparison.

B Match dispersion

This appendix derives the effective exponent (14) used when `--match-dispersion` $\epsilon > 0$, and states exactly what is averaged. The quantity averaged is the kernel weight for one template pair (j, k), $e^{-(\rho^2/2)D'}$, over a scalar Gaussian offset in the signal's distance to template k; the effective exponent is minus the logarithm of that expectation. The same g_{eff} is used in the Z pass, so Z_j is the sum of the averaged weights. Along the direction toward template k the displaced mismatch is

$$D' = \left(\sqrt{D} + \sigma\xi\right)^2, \qquad \xi \sim \mathcal{N}(0, 1), \qquad \sigma^2 = \epsilon\,(1 - D), \tag{43}$$

where the variance is scaled by $(1 - D)$ so that a displacement cannot change the distance to an antipodal template. Averaging the kernel over ξ uses the moment generating function of a noncentral χ^2: for $X = (\mu + \sigma\xi)^2$,

$$\mathbb{E}\left[e^{-sX}\right] = \frac{1}{\sqrt{1 + 2s\sigma^2}} \exp\left[-\frac{s\mu^2}{1 + 2s\sigma^2}\right]. \tag{44}$$

With $s = \rho^2/2$, $\mu^2 = D$ and $t \equiv 2s\sigma^2 = \rho^2\epsilon(1 - D)$, $-\ln \mathbb{E}[e^{-sX}]$ is the effective exponent (14). Its properties:

- $\epsilon = 0$ recovers the exact kernel;
- $D = 1$ gives $t = 0$ and $g_{\text{eff}} = \rho^2/2$, so unrelated templates keep full suppression at every ϵ;
- for small D and large ρ, $g_{\text{eff}} \to D/(2\epsilon)$ independent of ρ, so templates within $\sim \epsilon$ mismatch of the best remain indistinguishable at any SNR;
- g_{eff} is smooth and monotone in D, avoiding the banding produced by an earlier clamp form $\max(\sqrt{D} - \sqrt{\epsilon}, 0)^2$.

The suggested value is the mean fitting-factor mismatch of the bank plus the expected waveform-systematics mismatch.

Three things this is not, since the word marginalisation promises more than the code does:

- It is not a single displacement of the signal in parameter space applied consistently to every template. The offset ξ is drawn independently for each pair (j, k), and only its component along the direction from j to k is modelled; the perpendicular components of a real displacement are dropped, which is what "linearised on the match sphere" means.
- The variance $\epsilon\,(1 - D)$ is a modelling choice, not derived. It is chosen so that a template orthogonal to the signal, $D = 1$, cannot be moved by the displacement, and so that near the signal, where $D \to 0$, the offset in mismatch is of order ϵ.
- It averages the weight, not its logarithm. That is the right operation for a likelihood, but it means g_{eff} is not g evaluated at a blurred D; the $\frac{1}{2}\ln(1 + t)$ term is the difference.

References

[1] B. J. Owen, "Search templates for gravitational waves from inspiraling binaries: Choice of template spacing," Phys. Rev. D **53**, 6749 (1996).

[2] B. J. Owen and B. S. Sathyaprakash, "Matched filtering of gravitational waves from inspiraling compact binaries: Computational cost and template placement," Phys. Rev. D **60**, 022002 (1999).

[3] T. Tanaka and H. Tagoshi, "Use of new coordinates for the template space in a hierarchical search for gravitational waves from inspiraling binaries," Phys. Rev. D **62**, 082001 (2000).

[4] D. A. Brown, I. Harry, A. Lundgren, and A. H. Nitz, "Detecting binary neutron star systems with spin in advanced gravitational-wave detectors," Phys. Rev. D **86**, 084017 (2012).