A. Derivation of Equation 1

To avoid overloading notation, write $p := \mu_X$, $q := \mu_Y$. In matrix form, we have:

$$d_{\mathcal{N}}(X,Y) = \frac{1}{2}\min_{C\in\mathscr{C}(p,q)}\Big(\sum_{ijkl}|X_{ik}-Y_{jl}|^2 C_{kl}C_{ij}\Big)^{\frac{1}{2}}$$

Expanding the term inside the square root yields three terms. The first is the following:

$$\begin{aligned}\sum_{ijkl} X_{ik}^2 C_{kl}C_{ij} &= \sum_{ik} X_{ik}^2 p_k p_i = \sum_i p_i \sum_k X_{ik}^2 p_k \\ &= \sum_i p_i (X.^{\wedge}2 p)_i = \langle p, X.^{\wedge}2 p\rangle.\end{aligned}$$

Here the first equality followed by marginalization. Another term is as follows:

$$\begin{aligned}\sum_{ijkl} Y_{jl}^2 C_{kl}C_{ij} &= \sum_{jl} Y_{jl}^2 q_l q_j = \sum_l q_l \sum_j Y_{jl}^2 q_j \\ &= \sum_l q_l (Y.^{\wedge}2 q)_l = \langle q, Y.^{\wedge}2 q\rangle.\end{aligned}$$

The final term is the only one that depends on C:

$$\begin{aligned}-2\sum_{ijkl} X_{ik}Y_{jl}C_{kl}C_{ij} &= -2\sum_{il}(XC)_{il}(CY)_{il} \\ &= -2\langle XC, CY\rangle = -2\,\mathrm{tr}(C^T X^T CY).\end{aligned}$$

The final equality holds by the definition of the Frobenius product, and this concludes the derivation of Equation 1.

We further note that in the special case where X, Y are symmetric positive definite, we can take a Cholesky decomposition to write:

$$X = UU^T,\ Y = V^T V.$$

Then we have:

$$\begin{aligned}\mathrm{tr}(C^T X^T CY) &= \mathrm{tr}(C^T U^T U C V^T V) \\ &= \mathrm{tr}(V C^T U^T U C V^T) \\ &= \langle UCV^T, UCV^T\rangle = \|UCV^T\|^2,\end{aligned}$$

where $\|\cdot\|$ denotes the Frobenius norm. The function $C \mapsto \|UCV^T\|^2$ is now seen to be convex.

B. Proofs

Proof of Proposition 3. Let $A = A_{XY}$. Writing $A_s = \frac{1}{2}(A + A^*)$ for the *symmetrization* of A, we observe that

$$\begin{aligned}2\langle A_s C, C\rangle &= \langle AC + A^*C, C\rangle = \langle AC, C\rangle + \langle C, AC\rangle \\ &= \langle AC, C\rangle + \langle AC, C\rangle = 2\langle AC, C\rangle.\end{aligned}$$

The computation then agrees with the computation in the symmetric setting of [23] after replacing A with its symmetrization. □

Lemma 10. *Let (Z, ω_Z, μ_Z) be a finite measure network. Let $f \in L^2(Z^2, \mu_Z^{\otimes 2})$. For $t \in [0,1]$, define $\omega_t : (Z\times Z)^2 \to \mathbb{R}$ as*

$$\begin{aligned}\omega_t\left((z_1,z_2),(z_3,z_4)\right) = (1-t)&\omega_Z(z_1,z_3) \\ &+ t\omega_Z(z_2,z_4) + tf(z_2,z_4).\end{aligned}$$

Also let Δ denote the diagonal coupling between μ_Z and itself, i.e. the pushforward of μ_Z under the diagonal map $z \mapsto (z,z)$. Then we have:

$$(Z\times Z, \omega_t, \Delta) \cong^w (Z, \omega_Z, \mu_Z).$$

Proof. Consider the projection map $\pi : Z\times Z \to Z$ defined by $(z_1,z_2) \mapsto z_1$. It suffices to show that $\pi_\#\Delta = \mu_Z$ and $\|\pi^*(\omega_Z + tf) - \omega_t\|_\infty = 0$. For the first assertion, let $A \in \mathrm{Borel}(Z)$. Then we have:

$$\pi_\#\Delta(A) = \Delta(A\times Z) = \mu_Z(A).$$

For the second assertion, let $((z_1,z_2),(z_3,z_4)) \in (Z\times Z)^2$. Suppose also $z_1 = z_2$, $z_3 = z_4$. Then we have:

$$\begin{aligned}&\pi^*(\omega_Z + tf)\left((z_1,z_2),(z_3,z_4)\right) \\ &= \omega_Z(z_1,z_3) + tf(z_1,z_3) \\ &= \omega_t((z_1,z_1),(z_3,z_3)) \\ &= \omega_t((z_1,z_2),(z_3,z_4)).\end{aligned}$$

The conclusion follows because Δ assigns zero measure to all pairs (z,z') where $z \neq z'$. □

Proof of Proposition 5. Let $X = (X, \omega_X, \mu_X)$ be a finite measure network and let $f \in L^2(Z^2, \mu_Z^{\otimes 2})$ for some $Z \in [X]$. We wish to derive a condition which guarantees that

$$\gamma(t) := [Z, \omega_Z + f, \mu_Z]$$

is a geodesic defined on $[0,1]$. For any t, $(Z, \omega_Z + tf, \mu_Z)$ lies in the same weak isomorphism class as

$$(Z\times Z, (1-t)\omega_Z + t(\omega_Z + f), \Delta),$$

where Δ denotes the diagonal coupling of Z with itself, as in Lemma 10. This is the general form of a geodesic given above (3). Moreover, $\gamma(0) = [X]$, by the definition of Z. It therefore suffices to find a condition on f which guarantees that Δ is an optimal coupling between Z and the measure network

$$Z_1 := (Z, \omega_Z + f, \mu_Z) \qquad (6)$$

Consider an arbitrary coupling μ of Z with Z_1. The squared distortion $\mathrm{dis}(\mu)^2$ is given by

$$\int_{(Z\times Z)^2} \left(\omega_Z(z_1,z_2) + f(z_1,z_2) - \omega_Z(z_3,z_4)\right)^2 \mu\otimes\mu,$$

where $\mu \otimes \mu$ is short for $\mu \otimes \mu((dz_1, dz_2), (dz_3, dz_4))$. We rewrite this as

$$\int_{(Z\times Z)^2} \Big\{ (\omega_Z(z_1,z_2) - \omega_Z(z_3,z_4))^2 \qquad (7)$$

$$+ 2\left(\omega_Z(z_1,z_2) - \omega_Z(z_3,z_4)\right) f(z_1,z_2) \Big\} \mu \otimes \mu$$

$$+ \int_{(Z\times Z)^2} f(z_1,z_2)^2 \mu \otimes \mu. \qquad (8)$$

By the fact that μ is a coupling of μ_Z with itself, the term in line (8) simplifies to

$$\int_{Z^2} f(z_1,z_2)^2 \mu_Z(z_1)\mu_Z(z_2).$$

On the other hand, this quantity is equal to the squared distortion $\mathrm{dis}(\Delta)^2$.

To guarantee that $\mathrm{dis}(\Delta) \leq \mathrm{dis}(\mu)$, it suffices that the bracketed term in (7) can be made non-negative. If each $|\omega_Z(z_1,z_2) - \omega_Z(z_3,z_4)|$ is zero, then ω_X is constant, in which case we immediately see that the bracketed term is nonnegative without restriction on f. Otherwise, let $\epsilon_{[X]}$ be one half of the infimal strictly positive value of $|\omega_Z(z_1,z_2) - \omega_Z(z_3,z_4)|$, ranging over all quadruples of points in Z. Since Z is weakly isomorphic to X, the images of ω_X and ω_Z are equal, and since X is finite these images are finite. It follows that the infimum $\epsilon_{[X]}$ is actually a minimum and is strictly positive. Under the assumption that $|f(z,z')| < \epsilon_{[X]}$ for each $z, z' \in Z$, it is straightforward to check that the bracketed term in (7) is nonnegative, and this completes the proof. □

Proof of Proposition 8. For simplicity, suppose that $S = \{Y\}$ contains a single finite network and write $F = F_S$. The general case follows by similar arguments. After alignment, we can assume that $X = (X, \omega_X, \mu_X)$, $Y = (X, \omega_Y, \mu_X)$ and that the diagonal coupling Δ is optimal.

Let $[f] \in T_{[X]}$. Once again, we assume for simplicity that f is defined on a finite measure network, which we may as well take to be X after realigning as necessary. The general case can be shown by adapting this specialized argument.

The first task is to compute the directional derivative $D_{[f]}F([X])$. For $t \geq 0$, let $X_t = (X, \omega_X + tf, \mu_X)$ and let μ_t denote an optimal coupling of X_t with Y such that that $\lim_{t\to 0^+} \mu^t$ is the diagonal coupling $\mu_X \otimes \mu_X$. Note that for each t, the quantity

$$\frac{1}{t}\left(F(\exp_{[X]}(t[f])) - F([X]) \right) \qquad (9)$$

is upper bounded by

$$\frac{1}{t}\left(\mathrm{dis}(\mu_t)^2 - \mathrm{dis}(\mu_X \otimes \mu_X)^2 \right).$$

It is a straightforward computation to show that this upper bound can be rewritten as

$$t \sum_{i,j} f(i,j)^2 \mu_X(i)\mu_X(j) \qquad (10)$$

$$+ 2\sum_{i,j} (\omega_X(i,j) - \omega_Y(i,j)) f(i,j)\mu_X(i)\mu_X(j).$$

On the other hand, (9) is lower bounded by

$$\frac{1}{t}\left(\mathrm{dis}(\mu_t)^2 - \mathrm{dis}_{X,Y}(\mu_t)^2 \right),$$

where $\mathrm{dis}_{X,Y}(\mu_t)$ is the distortion of μ_t treated as a coupling of X and Y. This simplifies to

$$t \sum_{i,j,k,\ell} f(i,j)^2 \mu_t(i,k)\mu_t(j,\ell) \qquad (11)$$

$$+ 2\sum_{i,j,k,\ell} (\omega_X(i,j) - \omega_Y(k,\ell)) f(i,j)\mu_t(i,k)\mu_t(j,\ell).$$

As $t \to 0^+$, quantities (10) and (11) both limit to

$$2\sum_{i,j} (\omega_X(i,j) - \omega_Y(i,j)) f(i,j)\mu_X(i)\mu_X(j),$$

and this therefore provides a formula for the directional derivative $D_{[f]}F([X])$.

Finally, we note that

$$2\sum_{i,j} (\omega_X(i,j) - \omega_Y(i,j)) f(i,j)\mu_X(i)\mu_X(j)$$

$$= \langle [f], \nabla F([X]) \rangle_{[X]}$$

if we take $\nabla F([X])$ to be represented by the matrix

$$(\nabla F(X))_{ij} = 2\left(\omega_X(x_i,x_j) - \omega_Y(y_i,y_j)\right),$$

which is the claimed form for this specific example. The general formula (for S of larger cardinality) is derived by linearity. □

C. Support sizes for optimal couplings

The benefit of our representation of geodesics between measure networks is the empirical observation that (approximations of) optimal couplings tend to be sparse. This allows a geodesic between measure networks X and Y to be represented in a much smaller space than the naive requirement of size $|X| \cdot |Y|$. We have observed that it is more typical for the representation to require size which is linear in $|X| + |Y|$. Experimental evidence for this observation is provided in Figures 9 and 10.

There is also theoretical evidence for the observed small support size phenomenon. In [7, 6] the authors show that random quadratic programming problems tend to have sparse

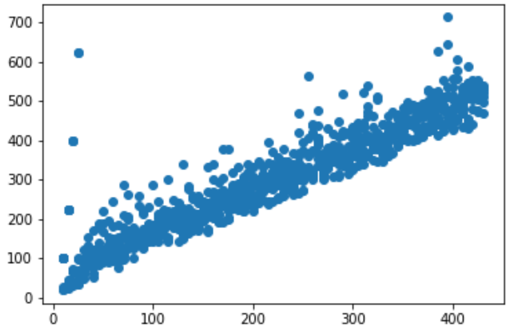

Figure 9. Support sizes for random measure networks. In each trial, a pair of Gaussian iid random weight matrices of size n is drawn. The optimal coupling for the uniformly weighted networks is computed and its support size is plotted against n. In general, the support size grows linearly.

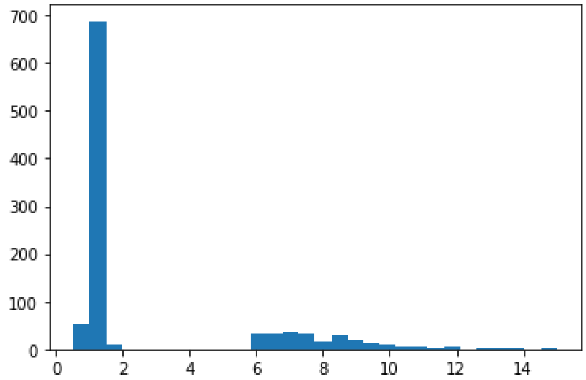

Figure 10. Support sizes for real networks. In each of 1000 trials, a random pair of graphs from the IMDB-BINARY graph classification benchmark dataset is chosen. The optimal coupling between their shortest path distance matrices (with uniform weights on the nodes) is computed. This histogram shows the distribution of support size divided by the sum of sizes of the graphs being compared. In general, the support size is a small multiple of the sum of graph sizes.

solutions with high probability. The setting of these articles is not exactly the one considered here (they use symmetric quadratic forms and optimize over the standard simplex) and it remains an open problem to give theoretical probabilistic guarantees for sparsity in the GW setting. Moreover, it would be interesting to get results for cost matrices with more realistic structures; e.g. binary matrices representing random directed adjacency matrices.

D. Support sizes for the iterative averaging scheme

Practical computation of Fréchet means as described in the main text comes with the standard challenges of nonconvex optimization: the gradient descent for finding optimal couplings may get stuck in bad local minima, and this in turn may propagate into poor computation of Fréchet means. Empirically we found that using a schedule for adjusting the gradient step size, i.e. using full gradient steps at the beginning and then using backtracking line search with Armijo conditions [21] often worked well. Accelerating the gradient descent using the momentum method also works well.

One aspect of the convergence problem is the size of the blowups needed to take discrete steps along the gradient flow of the Fréchet functional. Towards characterizing the classes of networks for which this problem is more or less difficult, we set up the following experiments. First we generated networks X_1, X_2 with random weight matrices generated using Python's `numpy.random.rand` function. We equipped these networks with uniform probabilities. Next we wrote $X_j = Y_j + D_j$ for $j = 1, 2$, where D_j consisted of the diagonal part of X_j, and Y_j had zero diagonal. Next we wrote $X_j^{(\alpha)} := Y_j + \alpha D_j$ for $\alpha \in \{0, 0.1, 0.2, \ldots, 1\}$. For each α, we set $\mathcal{X}^{(\alpha)} := \{X_1^{(\alpha)}, X_2^{(\alpha)}\}$ and computed the Fréchet mean of each $\mathcal{X}^{(\alpha)}$ using 100 randomly generated initial seed networks. We repeated this procedure in the cases where the X_j were both 10-node networks and where X_1 had 8 nodes, and X_2 had 10 nodes. Finally, we repeated this entire procedure after initially symmetrizing the X_j. The average sizes of the iterates are plotted against α in the left panel of Figure 11. The shading represents the standard deviation for each curve. First we note that as the diagonal terms are gradually added in, the sizes of the Fréchet mean iterates grow rapidly. This suggests that when preprocessing data for the Fréchet averaging procedure, it is helpful to use a scheme which enforces zero diagonals. The second observation is that there is some extra blowup that happens when averaging over a list of networks with different sizes. This is expected, as the optimal couplings between such networks cannot be permutation matrices, and hence some blowup is necessary.

Another interesting observation is that the level of asymmetry does not seem to affect the sizes of the iterates. However, asymmetry does affect the final Fréchet loss value at convergence. To test this effect, we generated matrices X_j as above and decomposed them into symmetric and antisymmetric parts: $X_j = S_j + A_j$. Next we chose α as above and considered the networks $Z_j^{(\alpha)} := S_j + \alpha A_j$. For each α, we set $\mathcal{Z}^{(\alpha)} := \{Z_1^{(\alpha)}, Z_2^{(\alpha)}\}$ and computed the Fréchet mean of each $\mathcal{Z}^{(\alpha)}$ using 100 randomly generated initial seed networks. We repeated this experiment for the cases where both X_j had 10 nodes, and where X_1 had 8 nodes and X_2 had 10 nodes. The values of the final Fréchet loss are plotted against α in the right panel of Figure 11. We observe that the final Fréchet loss increases with asymmetry, which suggests that the Fréchet function becomes more nonconvex with increasing asymmetry.

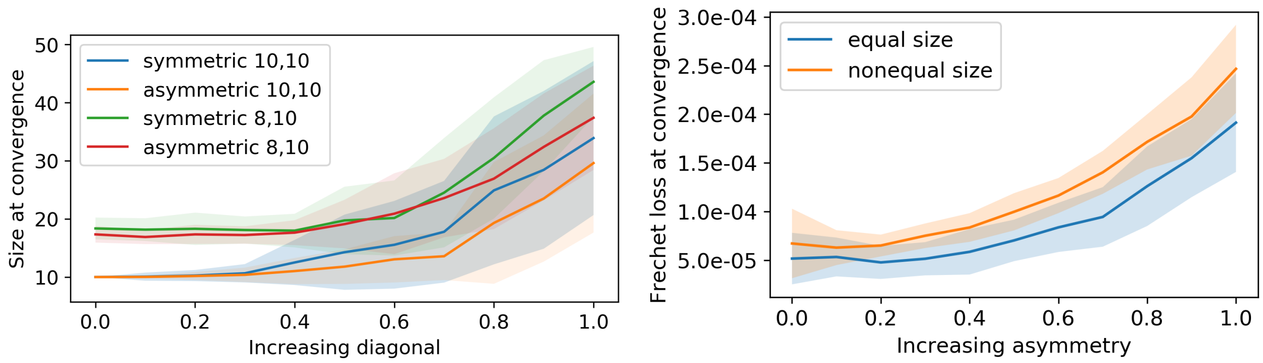

Figure 11. **Left:** The sizes of the iterates for the Fréchet mean procedure depend on the diagonal entries of the network weight matrices. However, these sizes are not influenced by the level of asymmetry in the matrices. **Right:** The values of the Fréchet loss function at convergence rise with increasing asymmetry of the network weight matrices.

These observations point to the following open questions:

- Can one place quantitative bounds on the rate of expansion of the Fréchet mean iterates as a function of the diagonal values of weight matrices?
- Can one adapt methods such as graduated nonconvexity to improve convergence for asymmetric networks, in the sense of "graduated asymmetry"?

To perform averages for networks with nonzero diagonal while circumventing the problem of expanding matrices, we adopted a simple—albeit Procrustean—method for restricting this expansion. This method has its own interesting application for *network compression*, and we detail it next.

D.1. Network compression

Let X, Y be finite networks, and let $\hat{X}$, $\hat{Y}$ denote their alignments. The aligned networks could, a priori, be larger in size than X and Y. Thus if the alignment is iterated, as would be the case in computing Fréchet means, we could have unbounded blowups in the sizes of these matrices. To prevent this situation, we pose the following question. Suppose $|X| < |Y|$. *What is the projection of the vector* $\omega_{\hat{Y}} - \omega_{\hat{X}}$ *onto the space of* $|X| \times |X|$ *vectors?* Let v denote this projection. Geometrically, we expect that $(X, \omega_X + v, \mu_X)$ is a good $|X|$-node representative of Y. Practically, we can take the average of (X, ω_X, μ_X) and $(X, \omega_X + v, \mu_X)$ without any expansion and expect this object to be an approximate average of X and Y.

We adopt the following simple method to obtain a low-dimensional representation of the tangent vector $\nu := \omega_{\hat{Y}} - \omega_{\hat{X}}$. Following the notation used in Definition 1, write $\hat{X} = X[\mathbf{u}]$. Recall that $\omega_{X[\mathbf{u}]}((x,i),(x',j)) = \omega_X(x,x')$. Define the $|X| \times |X|$-dimensional vector v as follows: for any $x, x' \in X$,

$$v(x,x') := \frac{\sum_{i=1}^{u_x}\sum_{j=1}^{u_{x'}}(\omega_{\hat{Y}} - \omega_{\hat{X}})((x,i),(x',j))}{u_x \cdot u_{x'}} \\ = \frac{\sum_{i=1}^{u_x}\sum_{j=1}^{u_{x'}}\omega_{\hat{Y}}((x,i),(x',j))}{u_x \cdot u_{x'}} - \omega_X(x,x').$$

Here we overload notation slightly to write $\omega_{\hat{Y}}((x,i),(x',j))$, but this is well-defined because $\hat{Y}$ is aligned to $\hat{X}$ and (x,i) is just an index.

To understand this construction, note that the elements of the tangent vector ν admit the following interpretation: ν_{pq} is just the difference $-\omega_{\hat{X}}(x_p, x_q) + \omega_{\hat{Y}}(y_p, y_q)$, i.e. it measures the change in the network weight from x_p to x_q when transferring from $\omega_{\hat{X}}$ to $\omega_{\hat{Y}}$. Here x_p, x_q are just indices of elements in $\hat{X}$. In the metric space setting, this quantity is exactly the change in distance between x_p and x_q that one would observe by following the optimal transport map $\hat{\mu}$ between $\hat{X}$ and $\hat{Y}$. Intuitively in the metric setting, points which start nearby and end nearby under the map $\hat{\mu}$ correspond to similar tangent vector entries.

Under this interpretation, the vector v simply averages out the changes that occur within and between blocks of $X[\mathbf{u}]$ when passing from $\omega_{\hat{X}}$ to $\omega_{\hat{Y}}$. Note in particular that $(X, \omega_X + v, \mu_X)$ gives us a *compressed representation* of Y. This is illustrated in Section 4.4.

Remark 11. The averaging method of [23] proceeds by fixing a size for the requested Fréchet mean and then performing an alternating optimization. This suggests the following open question: Is there a variant of the "compressed log map" approach outlined above that agrees with the method in [23]?

E. Algorithms

We now present pseudocode for our methods. Algorithm 1 serves as a placeholder; it can be computed using gradient descent [23] and is implemented in the Python Optimal Transport Library [12].

Algorithm 1 Compute minimizer of the GW functional

```
function optCoup($A, B, a, b$)
    // $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{m \times m}$. $a, b$ probability vectors
return $C$                                  ▷ $n \times m$ optimal coupling
end function
```

Algorithm 2 Computing the log map

```
function logMap($A, B, a, b$)
    // $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{m \times m}$. $a, b$ probability vectors
    // Lift geodesic from $A$ to $B$ to tangent vector based at $A$

    Initialize $splitData$ = []                 ▷ store metadata
    $C$ = optCoup($A, B, a, b$)
    Find rows, columns of $C$ with multiple nonzeroes
    Store indices in $splitData$
    Blow-up $A, B, a, b, C$ according to $splitData$
    $C = (C! = 0)$      ▷ convert $C$ to permutation matrix
    $B = C * B * C^T$                           ▷ align $B$ to $A$
    $v = -A + B$                                ▷ tangent vector
    return $A, a, v, splitData$
end function
```

Algorithm 3 Computing the Fréchet gradient

```
function frechetGrad($AList, aList, A, a$)
    // list of networks and a seed network
    Initialize $tanVec$ = []              ▷ list of tangent vectors
    $n$ = number of networks in $AList$
    $C$ = optCoup($A, B, a, b$)
    for $j = 0, \dots, n-1$ do
        $A, a, v, sD$ = logMap($AList[j], aList[j], A, a$)
        // $A, a$ may be blown-up at each step
        Use $sD$ to blow-up rows of $tanVec$ elements to be compatible with the newly blown-up $A$
        Append $v$ to $tanVec$
    end for
    $g = sum(tanVec)/n$                   ▷ Fréchet gradient
    return $g$
end function
```