Probe Selection for Imputation of Gene Expression Measurements

Yoni Donner, Ting Feng, Christophe Benoist, Daphne Koller

Supplementary Information

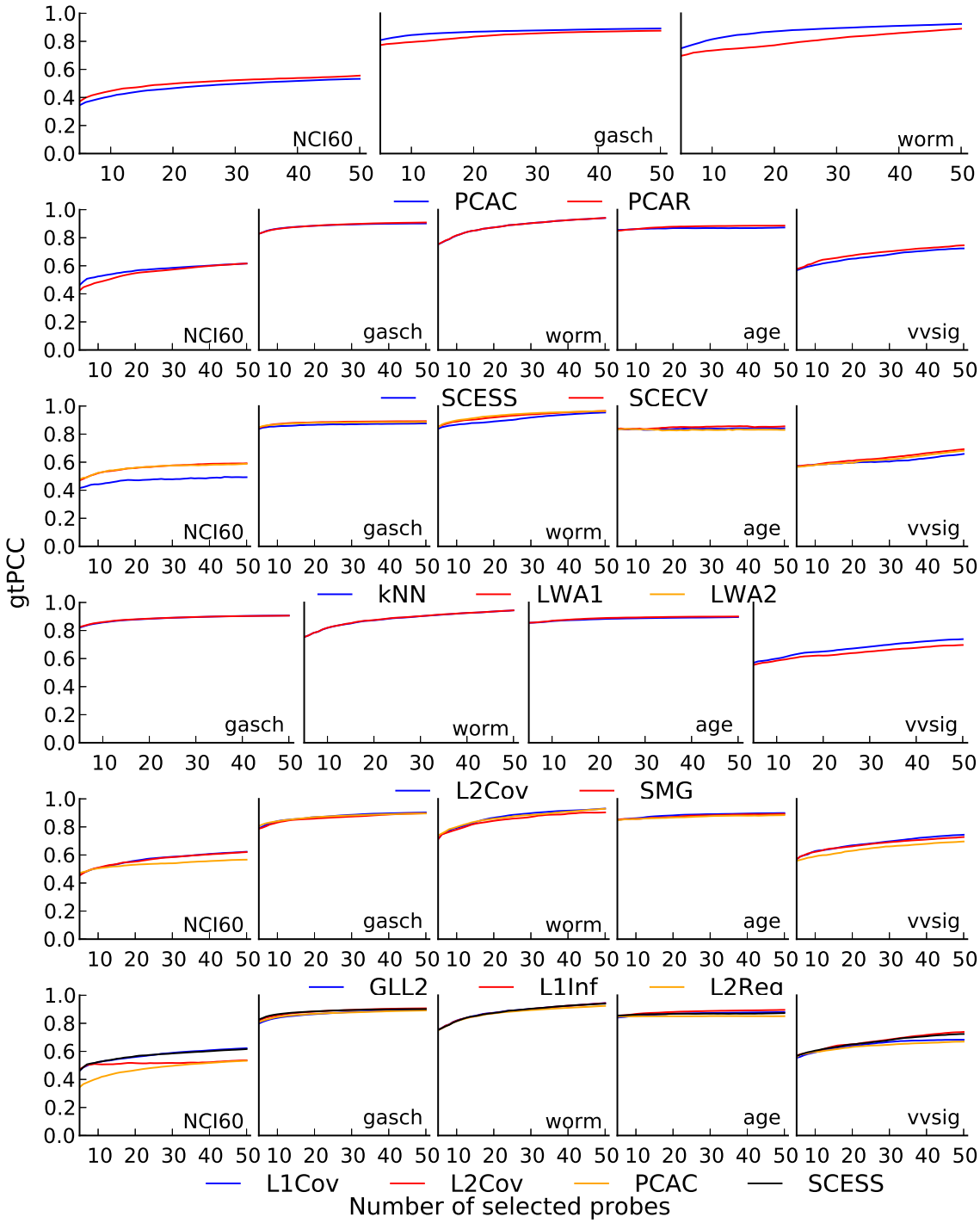
Supplementary Figures 1–8

Supplementary Table 1

Supplementary Results

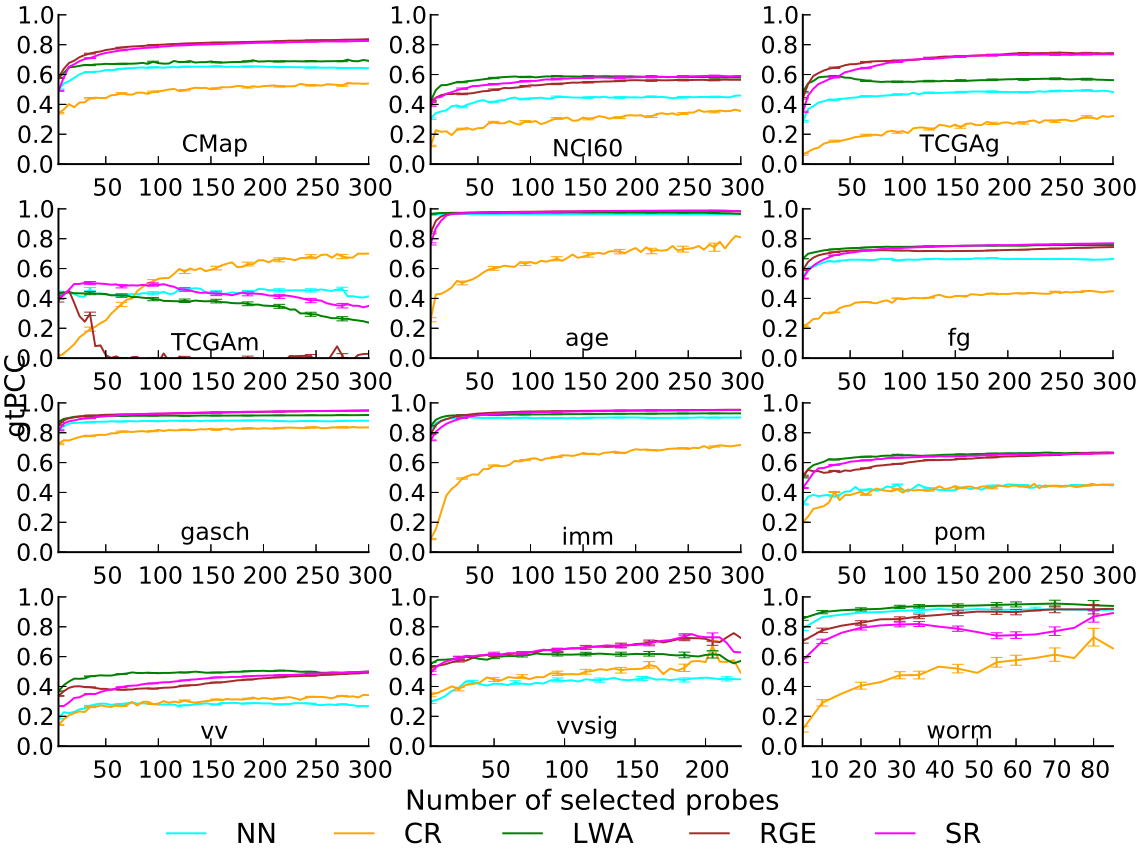
Supplementary Note: Complexity Analysis and Full Mathematical and Implementation Details

Supplementary Figure 1



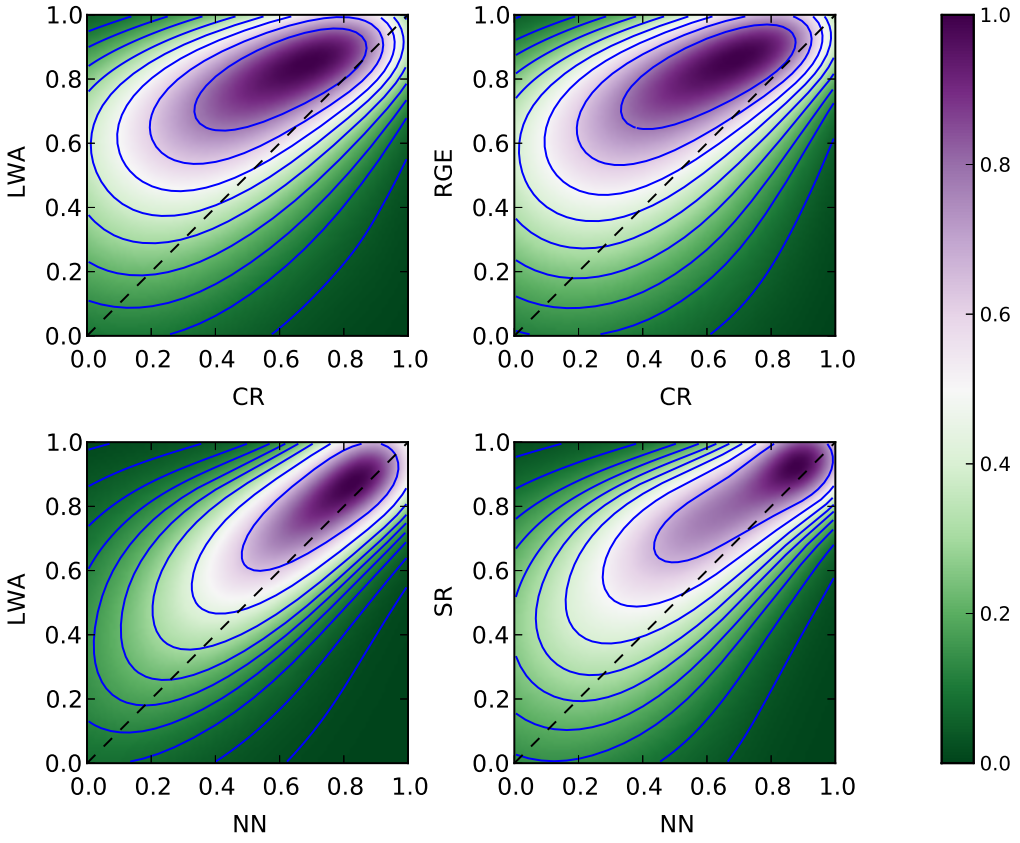
Supplementary Figure 1: Initial filtering of methods in multiple comparisons within method families on several datasets. From top to bottom: performance of the two PCA variants; performance of the two shrinkage-based covariance estimation variants; performance of kNN and the two LWA variants; performance of the RGE variants L2Cov and SMG; performance of the regression variants of GR (L2Reg) and SR (L1Inf and GLL2); performance of the final candidates to be used as the RGE representative method.

Supplementary Figure 2



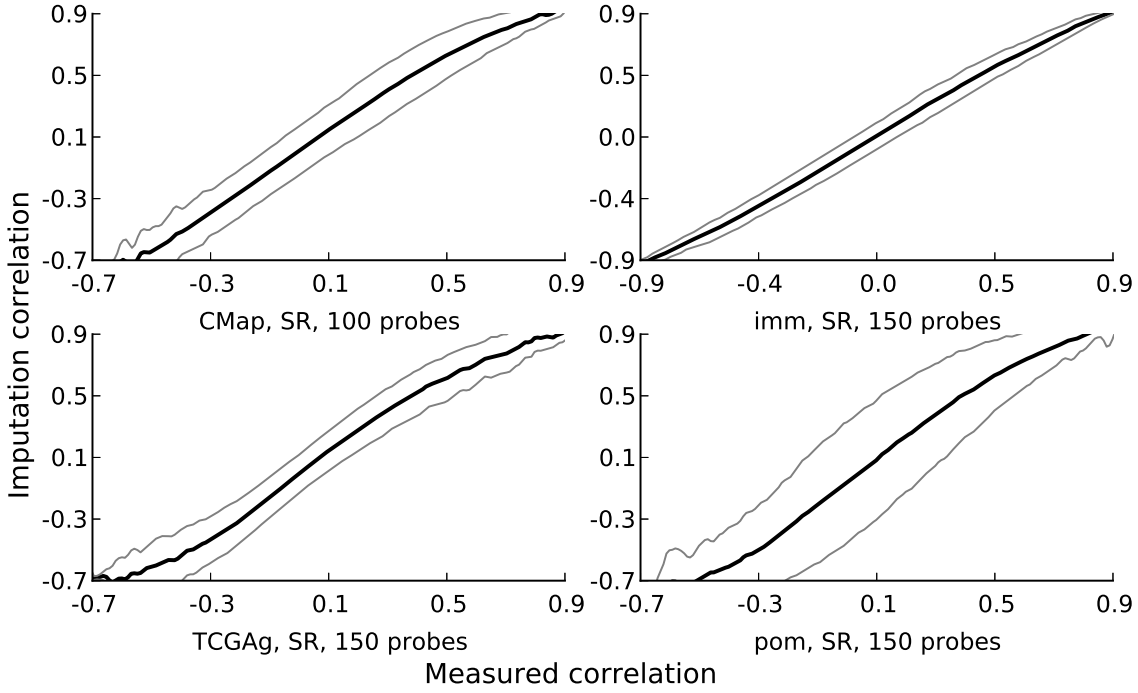
Supplementary Figure 2: gtPCC on all 12 datasets (y axis) as a function of the number of selected probes (x axis).

Supplementary Figure 3



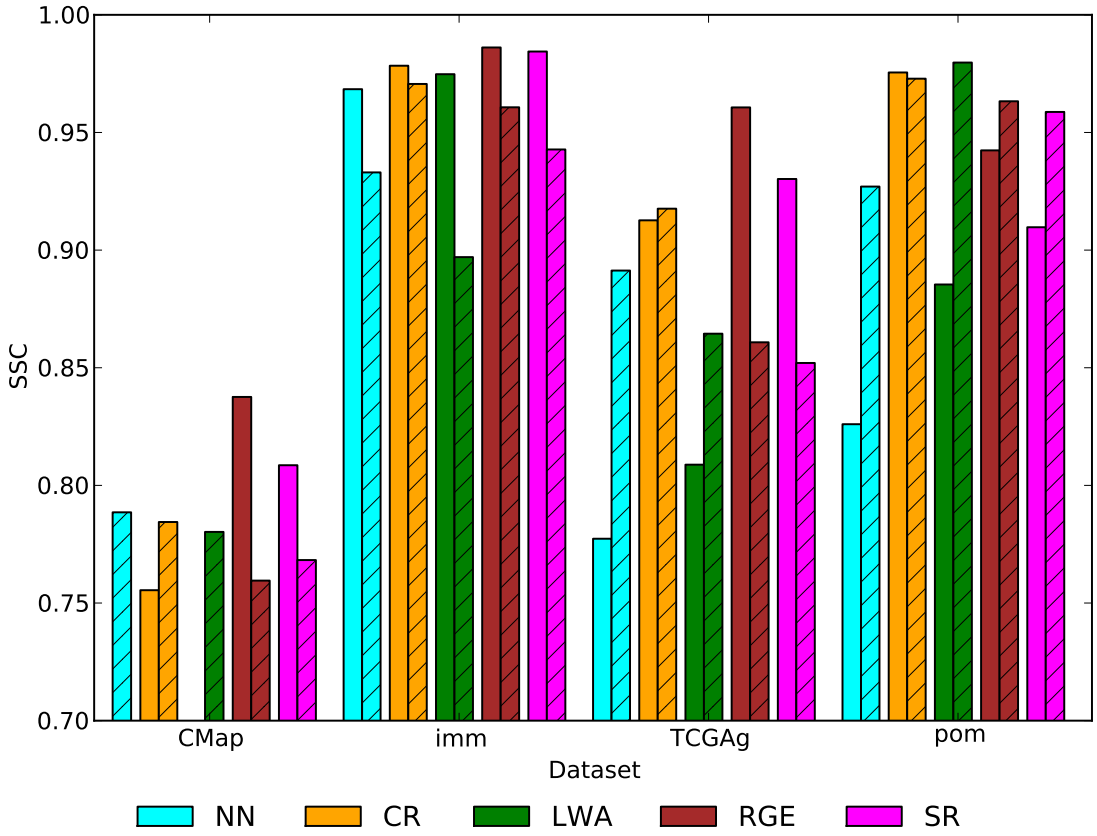
Supplementary Figure 3: Pearson correlation coefficients for individual probes using PSI methods (y axis) vs. baselines (x axis).

Supplementary Figure 4



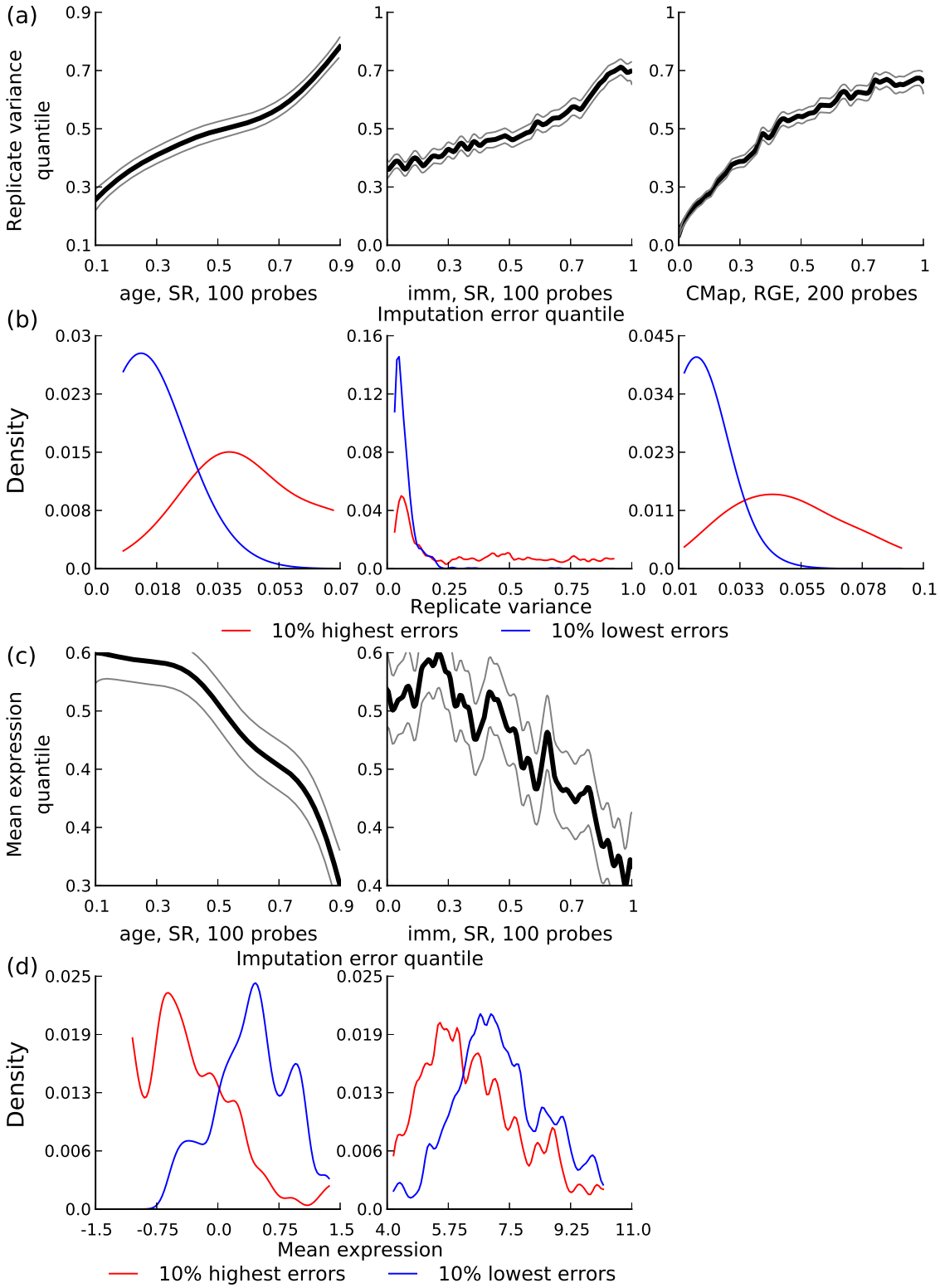
Supplementary Figure 4: Mean and 95% confidence intervals for correlation coefficients between imputations in smoothed windows around the correlation coefficients within the measured data, for the datasets and methods that were analyzed for their PPC values in the main text.

Supplementary Figure 5



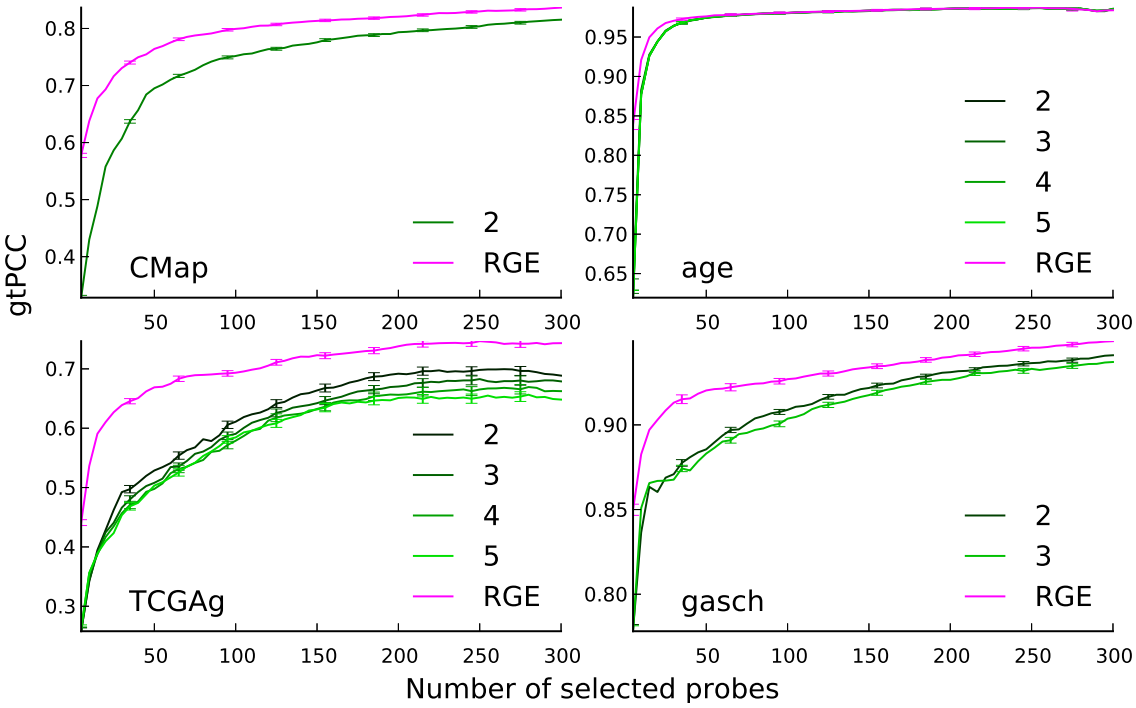
Supplementary Figure 5: Sample-sample correlations (SSC, see text) and tagSSC (textured bars, see text) on several datasets.

Supplementary Figure 6



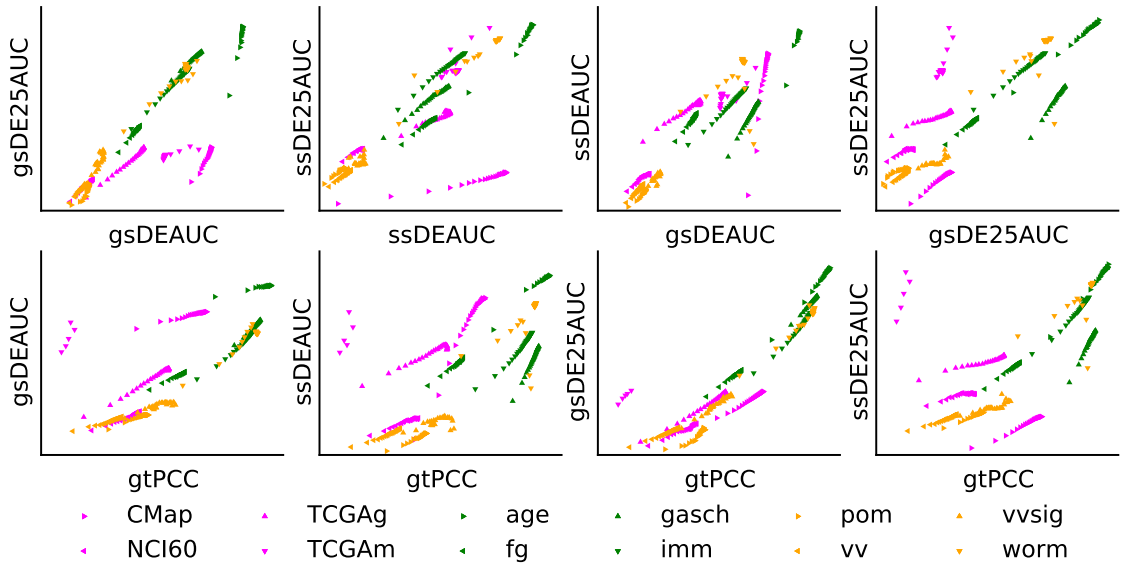
Supplementary Figure 6: Analysis of imputation error by variance of biological replices and absolute expression. (a) Mean quantile of replicate variance as a function of quantile of imputation error. (b) Density plots of biological replicate variance for the worst and best 10% probes by imputation error. (c) Mean absolute expression levels quantile by imputation error quantile. (d) Density plots of mean absolute expression levels for the best 10% and worst 10% probes by imputation accuracy.

Supplementary Figure 7



Supplementary Figure 7: Performance of the mixture model RGME with varying numbers of mixture components, also compared with the mixture-free RGE.

Supplementary Figure 8



Supplementary Figure 8: Comparisons of pairs of metrics, with each dataset marked by a different shape and color. A monotonic relationship within a dataset implies that both metrics are equivalent in terms of raking the different methods, and manifests as a group of points in the same shape and color such that a point which is to the right of another point is also above it.

Supplementary Table 1

Metric	Predictor	LWA	RGE	SR
Relative performance	SPRL	-0.740	0.717	0.567
	Samples	-0.336	0.484	0.438
	Selected probes	-0.351	0.468	-0.065
	Target probes	0.599	-0.225	-0.562
	Selected to target probe ratio	-0.748	0.549	0.46
	Linearity	-0.243	0.463	0.266
gtPCC	SPRL	0.485	0.735	0.743
	Samples	0.239	0.533	0.544
	Selected probes	0.046	0.196	0.163
	Target probes	-0.227	-0.481	-0.389
	Selected to target probe ratio	0.291	0.581	0.506
	Linearity	0.708	0.759	0.672

Supplementary Table 1: Spearman correlations between the components of SPRL and the relative and absolute (as measured by gtPCC) performance of LWA, RGE and SR.

Supplementary Results

Initial method selections

We developed many selection and imputation methods, based on established statistical methods and novel ideas. Some of our methods are similar in some aspects to other methods previously used for imputation but all were developed from scratch based on a unified framework of examining the joint selection and imputation problem.

We performed an initial evaluation of all methods using a more limited selection of data sets in order to first weed out methods that are strictly dominated by others, and later do an in-depth analysis of only the remaining methods.

PCA variants

We implemented two probe selection methods that are based on PCA: PCAC, which uses probabilistic PCA to estimate a covariance matrix and performs selection and imputation under the framework of Regularized Gaussian Estimation models; and PCAR, which uses PCA Regression directly. Since PCAC dominates (**Supplementary Fig. 1**, first row) and its running time is lower as well, we did not use PCAR is further experiments.

LWA with and without probe-specific kernel width

We compared the performance of Locally Weighted Averaging with and without probe-specific kernel width, and compared both with the nonparametric kNN method, with k chosen by cross-validation. Both LWA variants are better than kNN, and using probe-specific kernel width does not improve the imputation accuracy (**Supplementary Fig. 1**, third row). For this reason, we included only the more intuitive single-kernel-width LWA variant in the main comparison.

Comparison between linear regression methods

We implemented three methods based on regularized linear regression for imputation, and differ in the probe selection stage. GR (L2Reg) uses a greedy selection strategy whereas the two variants of SR involve variable selection and accuracy of imputation in a single objective. The SR variants dominate GR (**Supplementary Fig. 1**, fifth row). The two SR variants show very similar performance, so we picked the one based on the $L_{1,\infty}$ norm since it provides a stronger separation between the goals of variable selection and shrinkage. Theoretically, too much shrinkage could have a negative impact on the selection process.

Comparison between L2Cov and SMG

The two RGE variants L2Cov and SMG both estimate a covariance matrix by maximizing the data likelihood minus an L_2 penalty term on the entries of the inverse of the covariance matrix. L2Cov uses a uniform regularization coeffcient for all the entries in the precision matrix, while SMG uses an entry-specific coefficient which is based on the correlation between the two probes. L2Cov solves the problem in closed form using a single eigen-decomposition, while SMG uses an iterative gradient-based process, so L2Cov is far faster. The performance is of these two methods is nearly identical (**Supplementary Fig. 1**, fourth row) and the running time of SMG is much longer than that of L2Cov, so we evaluated only L2Cov on the remaining data sets.

RGE variants

We implemented several different methods for estimating covariance matrices for use with the RGE method. Although once the covariance matrix has been estimated, it is used within the

same framework, the estimation methods themselves differ in their accuracy and time and space requirements. For this reason we compared them first on a subset of the data sets to evaluate only the best-performing methods on all the data sets. The L_1-regularized precision matrix estimation method (L1Cov) and Soft Modular Gaussian (SMG) method both perform well in terms of accuracy (**Supplementary Fig. 1**, sixth row), but they both require multiple costly matrix inversions or eigen-decompositions of large $(m \times m)$ matrices, and since they do not seem to perform better overall than Shrinkage Covariance Estimation (the SCESS and SCECV variants) or L_2-regularized precision matrix estimation (L2Cov), which have far lower running times, we did not use them on the other data sets. Similarly, we chose not to use Probabilistic PCA (PCAC) on the other data sets, since its performance (imputation error) was worse than the other methods. The two methods of choosing α for the Shrinkage Covariance Estimation (SCESS and SCECV) method proved almost identical (**Supplementary Fig. 1**, second row).

SCESS and L2Cov are nearly identical in terms of accuracy and complexity. We chose to use the L2Cov variant as representative of the RGE method since it is slightly clearer theoretically, with a simple intuitive understanding using the well-studied eigenvalue decomposition. For data sets with too many probes to run the full Gaussian estimation, the only Gaussian method that was tested was MD, which is feasible since it assumes a modular structure in the covariance matrix, which greatly reduces computational and memory demands.

Identifying differential expression

To evaluate the abilities of our methods to identify differential expression, we used four related metrics. All four metrics are based on defining a differentially expressed subset of the held-out probes and evaluating the ability of the imputations to identify these probes by ranking imputations by distance from the mean, generating ROC curves and summarizing them using the area under curve (AUC).

We developed metrics both in probe-space, identifying which probes are differentially expressed in a given sample, and in sample-space, identifying in which samples is a given probe differentially expressed. For choosing the "true" subset of differentially-expressed probes, we used both absolute and relative criteria. The absolute criterion is a twofold or larger change in expression from the mean value. The relative criterion in being in the top 25%, measured by distance from the mean. For probe-space, the top 25% is relative to other probes in that sample, and for sample-space, the top 25% is relative to the expression values of that probe in other samples.

The four metrics are therefore: absolute in probe-space (GSDEAUC), relative in probe-space (GSDE25AUC), absolute in sample-space (SSDEAUC), relative in sample-space (SSDE25AUC).

Our experiments show that the four metrics are almost equivalent in terms of ranking, preserving the order between methods and number of selected probes within a dataset (**Supplementary Fig. 8**, top). Furthermore, they are also almost equivalent with the gtPCC metric (**Supplementary Fig. 8**, bottom), implying that our conclusions from the comparative analysis using gtPCC can be applied to the ability to detect differential expression as well.

For the main analysis of performance, we chose the metric of identifying which probes are differentially-expressed in a new sample, using the absolute criterion, since this combination is most commonly used in practice.

Density heatmaps for baseline vs advanced methods

As another form of validation, we compared the imputations of the three leading methods against the baselines on an individual probe level. The improvements of the advanced methods (**Supplementary Fig. 3**) are robust across the full range of expression values.

Mixture model results

The main limitation of the RGE, GR and SR methods is that the resulting imputations are all linear functions of the tag probes. To overcome this limitation, we developed a mixture model (RGME) on top of RGE, where the full expression data is modeled as a mixture of Gaussians, and therefore no longer linear. The model is computationally difficult to train (using the EM algorithm) and we first wanted to see if it results in any improvements at all in the imputation accuracies beyond the linear models. If the data is sufficiently well-modeled by linear models, the added number of parameters might actually hurt performance.

The performance of RGME did not significantly exceed that of the mixture-free equivalent RGE on any dataset with any number of mixture components (**Supplementary Fig. 7**), so we did not include it in our in-depth method analysis.

SSC results

Similarly to clustering probes, clustering samples is another common analysis done on gene expression data. We used the sample-sample correlation (SSC) metric, defined analogously to PPC, using the target probes to compute the sample correlations (to avoid biasing the result with the selected probes, which are always measured). An interesting baseline for this metric is the similarity between the true sample-sample correlation matrix based on target probes and the sample-sample correlation matrix computed using just the selected probes. We refer to this baseline as "tagSSC".

Several observations can be made about the SSC results (**Supplementary Fig. 5**): first, as with the other metrics, the linear methods SR and RGE dominated on high-SPRL datasets, and the imputations using these methods ranked higher than the corresponding tagSSC values. This result suggests that the imputations managed to accurately predict expression patterns that are not directly reflected in the tag probes. However, on pom, with its much lower SPRL value, the tagSSC values were higher. Second, CR performed very well by this metric, and its SSC and tagSSC values were nearly identical. This is not surprising, since its imputations are all taken from the expression levels of the selected probes. Therefore, the difference between SSC and tagSSC for CR is only due to different cluster sizes: each selected probe contribution to the tagSSC is identical, but its contribution to the SSC is proportional to the size of the cluster of which it is the representative. Third, LWA performed poorly on this metric. This may be explained by its global nature, which strongly limits its ability to capture complex patterns in the sample-sample correlation matrix, that are due to distinct patterns of differential expression in different probe subsets, a strongly local phenomenon that is well-modeled by the parametric methods.

Analysis of probes by imputation accuracy

Since a high imputation accuracy on average does not guarantee that all target probes are imputed with high accuracy, and since some probes might be more important than others, we did an analysis of probes by imputation error to make sure that the high overall accuracy is not due to high accuracy on many unimportant probes while more important probes are poorly-imputed.

We first examined imputation error vs. the variance in biological replicates (**Supplementary Fig. 6a**), which reflects mostly measurement error, for the three datasets for which we had replicated (CMap, imm, age). High imputation error is strongly associated with high replicate variance, which suggests that the imputation errors of these probes are to a large degree due to unreliable measurements. The distribution of replicate variance of the probes whose imputation errors are in the top 10% is shifted to the right compared to those probes whose imputation errors are in the bottom 10% (**Supplementary Fig. 6b**).

We next examined the absolute expression levels of probes compared to their imputation accuracy (**Supplementary Fig. 6c**) for two of the same datasets (the CMap data was centered at zero, so we did not have absolute expression levels). High imputation error is associated with

low absolute expression levels, which also supports the hypothesis that the imputation error is to a large degree due to unreliable measurements. The distribution of absolute expression levels is shifted towards lower absolute levels for probes in the top 10% worst imputed probes, as compared with the best 10% imputed probes (**Supplementary Fig. 6d**).

Finally, we carried out an analysis of Gene Ontology terms associated with the worst 10% imputed probes, and looked for terms that are underrepresented or overrepresented relative to all target probes. No terms were significant at $p < 0.05$ after Bonferroni correction.

The three analyses described above suggest that the less successful imputations are mostly due to unreliable measurements but not related to any functional classification.

Preservation of correlation structure across all correlation coefficients

PPC is a metric that compares the correlation structure within the observed data to the correlation structure within the imputations, and our analysis showed that the correlation structure was strongly preserved in the imputations. It is also of interest to verify that correlations are preserved not only in aggregate but across the whole range of correlation coefficients. We compared the average correlation coefficients between the imputed probe values in windows around each value of measured correlations (**Supplementary Fig. 4**), for the same datasets, methods and numbers of selected probes that were used to analyze PPC in the main text. The correlations within the imputed data were almost identical to the ground-truth correlations across the full range of correlation coefficients, suggesting that the imputations indeed correctly captured the correlation structure in the data.

Contributions of individual predictors to the predictive power of SPRL

SPRL predicts relative performance of the individual methods as well as absolute performance as measured by gtPCC. Since SPRL is the product of several factors, it is of interest to quantify the individual contribution of these factors. We computed Spearman correlation coefficients between each component of SPRL (as well as SPRL itself) and the relative and absolute performance of the three leading methods: LWA, RGE and SR. For relative performance, the ratio of selected to target probes was the strongest contributer, followed by the number of samples. For absolute performance, the strongest contributor was linearity, followed by the ratio of selected to target probes (**Supplementary Table 1**).

Supplementary Note

Complexity Analysis and Full Mathematical and Implementation Details

1 Complexity

In this section we briefly analyze the time and space complexity of the probe selection and covariance estimation methods.

1.1 Nearest Neighbor

For each new probe, updating the distance matrix for selected probes and for remaining probes takes $O(n^2)$. Finding the nearest neighbors takes $O(n)$ for each sample, so $O(n^2)$ total. Since there are $O(m)$ probes to test, adding a new probe takes $O(n^2 m)$ total operations, and for l probes we have $O(lmn^2)$.

Space complexity is simply $O(n^2)$ for the distance matrices.

1.2 k-Nearest Neighbors

As in NN, updating the distance matrices takes $O(n^2)$. Finding the k nearest neighbors for all n samples takes $O(k \log n)$. Computing the LOOCV errors for using $1, \ldots, k$ neighbors takes $O(nk^2)$. The total time complexity is, therefore, $O(lmn(n+k^2))$. The space complexity is the same as NN, $O(n^2)$.

1.3 Locally weighted averaging

As in NN, for each candidate new probe, updating the distance matrices and computing the weights takes $O(n^2)$. Computing the sum of squared imputation errors for each of the n samples takes $O\left(n \min\left(m, n\right)\right)$. If we test n_α different values for the kernel width parameter, the total time complexity is $O\left(lmn^2 \min\left(m, n\right) n_\alpha\right)$. The space complexity is $O\left(n^2\right)$ (maintaining the distance matrix).

The complexity using probe-specific kernel widths is the same, but we need to add the complexity of re-optimizing the kernel widths, which is $O\left(l^2 n^2 \min\left(m, n\right) n_\beta n_i\right)$, where n_β is the number of values for β_j tested, and n_i is the number of times we cycle through the kernel widths before convergence.

1.4 PCA Regression

When adding the $t+1$'th probe, we invert, for each $1 \leq k \leq \min(m, n-2)$ and $\lambda \in \Lambda$, $m-t$ matrices of size $k \times k$. We do this in $O\left(k^2\right)$ for each k and λ, so in total $O\left(mn_\lambda \min\left(n, m\right)^3\right)$. To choose all l probes and their corresponding values for k and λ, the overall time complexity is $O\left(lmn_\lambda \min\left(n, m\right)^3\right)$. The space complexity is $O\left(\min\left(n, m\right)^2\right)$.

1.5 L_2-regularized linear regression

When adding the t'th probe, the residual matrix is computed in $O(mtn)$ and the scores for all probes in $O(m^2n)$. Solving for the new λ and the new weights is done in $O(n_\lambda m^2 n)$ where n_λ is the number of values of λ evaluated. Since this is done l times, the overall time complexity is $O(ln_\lambda m^2 n)$. The space complexity is $O(lm)$ for the weight matrix.

1.6 GLL2 Regression

Updating an entire row in the weight matrix takes $O(mn)$. A single iteration of cycling through all weights takes $O(m^2 n)$. Let n_i denote the total number of iterations done when solving for the weights over all values of λ. Then the total time complexity of probe selection is $O(n_i m^2 n)$. Even when using warm starts n_i is likely to be quite large and at least on the order of l times a large constant. The space complexity is $O(m(n+l))$.

To solve for the weights using L_2-regularized linear regression, the time complexity is as for L2Reg, $O(ln_\lambda m^2 n)$ and the space complexity is $O(lm)$.

1.7 $L_{1,\infty}$ Regression

Updating an entire row in the weight matrix takes $O(mn + m\log m)$ time. A single iteration of cycling through all weights takes $O(m^2(n+\log m))$. As with the GLL2 method, let n_i denote the total number of iterations done when solving for the weights over all values of λ. Then the total time complexity of probe selection is $O(n_i m^2(n+\log m))$. Even when using warm starts n_i is likely to be quite large and at least on the order of l times a large constant. The space complexity is $O(m(n+l))$.

Solving for the weights using L_2-regularized linear regression takes $O(ln_\lambda m^2 n)$ time and $O(lm)$ space.

1.8 Gaussian methods

To add a new probe, we compute the full conditional covariance matrix given the current set. The total cost of inverting the submatrices $\Sigma_{\mathcal{S}^{(t)},\mathcal{S}^{(t)}}$ is $O(l^3)$, since this is done incrementally. Computing $\Sigma_{-\mathcal{S},\mathcal{S}}\Sigma_{\mathcal{S},\mathcal{S}}^{-1}\Sigma_{\mathcal{S},-\mathcal{S}}$ takes $O(m^2 t)$ (for iteration t), so in total $O(m^2 l^2)$. Since $l < m$, this term dominates the time of inverting the submatrices, and also the time of choosing a probe to minimize the remaining conditional variance. Hence, the total time complexity is $O(m^2 l^2)$. The space complexity is $O(m^2)$ for storing covariance matrices and inverses.

1.9 PCA Covariance Estimation

To choose the number of components, we compute the covariance matrix for each data subset (with one sample left out) and compute the likelihood of the remaining sample for $k = 0, \ldots, \min(m, n-2)$, which can be done efficiently by incremental updates in a total time of $O(m^2 n + mn^2)$. Generating the actual estimated covariance matrix takes $O(m^2 n)$. The total time complexity is therefore $O(m^2 n + mn^2)$ and the space complexity is $O(m^2)$.

1.10 Shrinkage Covariance Estimation

Estimating α takes $O(m^2 n)$ time and $O(m^2)$ space using SCESS. Using cross-validation, it takes $O(m^2)$ for each of the n samples, so $O(n_\alpha m^2 n)$ overall, where n_α is the number of values we evaluate for α.

1.11 L_2-regularized precision matrix estimation

We choose σ^2 by leave-one-out cross-validation. We first compute the SVD of $\boldsymbol{X}$ in $O\left(mn\min\left(m,n\right)\right)$. Then, for each value of σ^2 that we evaluate, we go over each of the n training samples and estimate the covariance matrix of the data with this sample held out. Using efficient rank-one updates instead of full recomputations, this requires $O\left(m^2\right)$ operations for each of the n samples, and overall $O\left(n_{\sigma^2}nm^2\right)$.

1.12 L_1-regularized precision matrix estimation

The time complexity is $O\left(m^3\right)$ for each iteration. With n_i iterations until convergence, this is $O(n_i m^3)$. With n_λ different values of λ to evaluate and n_s cross-validation splits, the overall time complexity is $O\left(n_s n_\lambda n_i m^3\right)$. The space complexity is $O\left(m^2\right)$.

1.13 Soft modular Gaussian

First, γ is chosen as in L2Cov, in $O\left(n_\gamma nm^2\right)$ time. Then, γ_1 and γ_2 are chosen with a cross-validation procedure similar to L1Cov above, and since the time and space complexity of estimating the precision matrix with L_1 and L_2 regularization are the same, this is done in $O\left(n_s n_{\gamma_2} n_i m^3\right)$ time and $O\left(m^2\right)$ space. Without choosing parameters by cross-validation, this is simply $O\left(n_i m^3\right)$ time.

1.14 Modular Decomposition

The time complexity of running K-Means is $O\left(n_i n_M nm\right)$ where n_i is the number of iterations before convergence. The space complexity is $O\left(\left(m+n_M\right)n\right)$. The time and space complexity of probe selection are given in Section 2.6.1 where the method is described.

1.15 Mixture of Gaussians

Let n_c be the number of components in the mixture. We first choose σ^2 using cross-validation. This step takes $O\left(n_{\sigma^2}nm^2\right)$ as described above for L2Cov. Then we use EM to learn the mixture model. We use n_r restarts and let n_i denote the number of iterations until convergence. Each iteration requires $O\left(n_c nm^2\right)$ for the E-step (computing $Q_i(c)$ for $i=1,\ldots,n$ and $c=1,\ldots,n_c$) and $O\left(n_c nm^2\right)$ for the M-step, since for each of the n_c mixture components we compute the SVD of a $m\times n$ matrix, modify the eigenvalues and compute the $m\times m$ covariance matrix. The total time complexity for the EM learning is $O\left(n_i n_c nm^2\right)$ and the space complexity is $O\left(n_c m^2\right)$.

For probe selection, adding the t'th probe requires computing $P_{\langle\mu_c,\Sigma_c\rangle}\left(X_{\mathcal{S}^{(t-1)}\cup\{j\},i}\right)$ for $j\notin\mathcal{S}^{(t-1)}$, $c=1,\ldots,n_c$ and $i=1,\ldots,n$. Each such computation can be done in $O\left(t^2\right)$, so overall $O\left(n_c nmt^2\right)$. Since we do this for $t=1,\ldots,l$, the overall time complexity for probe selection is $O\left(n_c nml^3\right)$. The space complexity is still $O\left(n_c m^2\right)$.

1.16 Summary

The following table shows the time and space complexity for all methods.

	Selection		Imputation	
Method	Time	Space	Time	Space
NN	$O(lmn^2)$	$O(n^2)$	$O(m+ln)$	$O(1)$
kNN	$O\left(lmn\left(n+k^2\right)\right)$	$O(n^2)$	$O(km+ln)$	$O(k)$
LWA1	$O\left(lmn^2\min\left(m,n\right)n_\alpha\right)$	$O(n^2)$	$O\left(nm\right)$	$O(n)$
LWA2	$O\left(ln^2\min\left(m,n\right)\left(mn_\alpha+ln_\beta n_i\right)\right)$	$O(n^2)$	$O\left(nm\right)$	$O(n)$
PCAR	$O\left(lmn_\lambda\min\left(n,m\right)^3\right)$	$O\left(\min\left(n,m\right)^2\right)$	$O\left(m\min\left(n,m\right)\right)$	$O\left(\min\left(n,m\right)^2\right)$
L2Reg	$O\left(ln_\lambda m^2n\right)$	$O\left(lm\right)$	$O(lm)$	$O(lm)$
GLReg	$O\left(n_im^2n\right)$	$O\left(m\left(n+l\right)\right)$	$O(lm)$	$O(lm)$
GLL2	$O\left(m^2n\left(n_i+ln_\lambda\right)\right)$	$O\left(m\left(n+l\right)\right)$	$O(lm)$	$O(lm)$
L1Inf	$O\left(m^2\left(n_in+n_i\log m+ln_\lambda n\right)\right)$	$O\left(m\left(n+l\right)\right)$	$O(lm)$	$O(lm)$
Gaussian models	$O\left(l^2m^2\right)$	$O\left(m^2\right)$	$O(lm)$	$O\left(m^2\right)$
PCAC	$O\left(mn^2+m^2n\right)$	$O\left(m^2\right)$	$O(1)$	$O(1)$
SCESS	$O\left(m^2n\right)$	$O\left(m^2\right)$	$O(1)$	$O(1)$
SCECV	$O\left(n_\alpha m^2n\right)$	$O\left(m^2\right)$	$O(1)$	$O(1)$
L1Cov	$O\left(n_sn_\lambda n_im^3\right)$	$O\left(m^2\right)$	$O(1)$	$O(1)$
L2Cov	$O\left(n_{\sigma^2}nm^2\right)$	$O\left(m^2\right)$	$O(1)$	$O(1)$
SMG	$O\left(n_\gamma nm^2+n_sn_{\gamma_2}n_im^3\right)$	$O\left(m^2\right)$	$O(1)$	$O(1)$
MD	$O\left(n_in_Mnm+mn_M+l^2\sum_{i=1}^{n_M}\left\|M_i\right\|^2\right)$	$O\left(\left(m+n_M\right)n+\sum_{i=1}^{n_M}\left\|M_i\right\|^2\right)$	$O\left(lm\right)$	$O\left(\left(m+n_M\right)n+\sum_{i=1}^{n_M}\left\|M_i\right\|^2\right)$
RGME	$O\left(nm\left(n_{\sigma^2}m+n_cn_im+n_cl^3\right)\right)$	$O\left(n_cm^2\right)$	$O\left(n_clm\right)$	$O\left(n_cm^2\right)$

2 Full mathematical and implementation details for all methods

2.1 Notation

Let $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ be the fully observed data matrix of m genes and n samples. The individual samples are $X_1,\ldots,X_n \in \mathbb{R}^{m\times 1}$. Individual gene expression values are denoted by $x_{j,i}$ (j'th gene in i'th sample). A new test sample is denoted by $Y \in \mathbb{R}^{m\times 1}$. Gene expression matrices and vectors corresponding to subsets $\mathcal{S}$ of genes are denotes by $\boldsymbol{X}_\mathcal{S}$ and $X_{\mathcal{S},i}$ respectively. For test samples, only $Y_\mathcal{S}$ is observed, so the full vector Y is completed by imputation.

2.2 k-Nearest Neighbors

2.2.1 Selection and imputation

For a new observed test vector $Y_\mathcal{S}$, find its k nearest neighbors among $\boldsymbol{X}_\mathcal{S}$:

$$\mathcal{I}\left[\boldsymbol{X},Y;\mathcal{S},k\right] = \arg\min_{\mathcal{I}\subseteq\{1,\ldots,n\}:|\mathcal{I}|=k}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{S}}\left(X_{j,i}-Y_j\right)^2$$

The imputation is then

$$Y = \frac{1}{k}\sum_{i\in\mathcal{I}[\boldsymbol{X},Y;\mathcal{S},k]}X_i$$

In words, we find the k nearest neighbors and predict their average.

Probe selection is done by greedy forward selection. The set $\mathcal{S}$ is constructed iteratively, where at each step a new probe is chosen to minimize the leave-one-out cross-validation (LOOCV) error, given the probes already chosen. Denote the LOOCV error with k nearest neighbors and the probeset $\mathcal{S}$ by $\mathcal{J}\left[\boldsymbol{X};\mathcal{S},k\right]$, then:

$$\mathcal{J}\left[\boldsymbol{X};\mathcal{S},k\right]=\sum_{i=1}^{n}\sum_{j\notin\mathcal{S}}\left(x_{j,i}-\frac{1}{k}\sum_{i'\in\mathcal{I}[\boldsymbol{X}_{-i},X_i;\mathcal{S},k]}x_{j,i'}\right)^2$$

Initially, $\mathcal{S}^{(0)}=\phi$. Then, in each new iteration, all probe candidates $j\notin\mathcal{S}^{(t)}$ are examined to minimize the LOOCV error:

$$\hat{j}_{\mathcal{S}^{(t)}}=\arg\min_{j\notin\mathcal{S}^{(t)}}\mathcal{J}\left[\boldsymbol{X};\mathcal{S}\cup\{j\},k\right]$$

Then, $\mathcal{S}^{(t+1)}=\mathcal{S}^{(t)}\cup\{\hat{j}_{\mathcal{S}^{(t)}}\}$.

A further improvement involves choosing $k\in\{1,\ldots,k_{\max}\}$ itself to minimize the LOOCV error:

$$\langle\hat{j}_{\mathcal{S}^{(t)}},\hat{k}\rangle=\arg\min_{j\notin\mathcal{S}^{(t)},1\leq k\leq k_{\max}}\mathcal{J}\left[\boldsymbol{X};\mathcal{S}\cup\{j\},k\right]$$

2.2.2 Implementation

A naive implementation to select l probes involves l iterations, where in each iteration $O\left(m\right)$ probes and $k_{\max}$ values of k are considered given the current set $\mathcal{S}^{(t)}$ to minimize the LOOCV error. Computing $\mathcal{I}\left[\boldsymbol{X}_{-i},X_i;\mathcal{S},k\right]$ requires the distances between $X_{\mathcal{S},i}$ and $X_{\mathcal{S},i'}$ to be computed for all $i'\neq i$, which is $O\left(|\mathcal{S}|n\right)$ and then sorted (we can ignore this cost). Computing the actual error for a given set of neighbors is $O\left((m-|\mathcal{S}|)k\right)$. Since we do this n times, the complexity for each pair $\langle\mathcal{S},k\rangle$ is $O\left(n^2|\mathcal{S}|+nmk\right)$. Since we consider $O\left(m\right)$ probes and $k_{\max}$ values of k, and we have l iterations, the overall complexity is $O\left(lnm^2k_{\max}^2+l^2n^2mk_{\max}\right)$.

To make this more efficient, we first note that $\|X-Y\|_2^2=X^TX+Y^TY-2X^T$, so

$$\begin{aligned}\sum_{j\notin S}\left(x_{j,i}-\frac{1}{k}\sum_{i'\in\mathcal{I}[\boldsymbol{X}_{-i},X_i;\mathcal{S},k]}x_{j,i'}\right)^2 &= X_{-\mathcal{S},i}^TX_{-\mathcal{S},i}-\\ &- \frac{2}{k}\sum_{i_1\in\mathcal{I}[\boldsymbol{X}_{-i},X_i;\mathcal{S},k]}X_{-\mathcal{S},i_1}^TX_{-\mathcal{S},i}+\\ &+ \frac{1}{k^2}\sum_{i_1,i_2\in\mathcal{I}[\boldsymbol{X}_{-i},X_i;\mathcal{S},k]}X_{-\mathcal{S},i_1}^TX_{-\mathcal{S},i2}\end{aligned}$$

and

$$\mathcal{J}\left[\boldsymbol{X};\mathcal{S},k\right]=\mathrm{Tr}\left(W\left[\mathcal{S},k\right]^TA\left[\mathcal{S}\right]\right)$$

where $A[\mathcal{S}]\in\mathbb{R}^{n\times n}$ is defined by $A[\mathcal{S}]_{i_1,i_2}=X_{-\mathcal{S},i_1}^TX_{-\mathcal{S},i_2}$, and $W\left[\mathcal{S},k\right]=I+U\left[\mathcal{S},k\right]+V\left[\mathcal{S},k\right]$, where $U\left[\mathcal{S},k\right]_{i_1,i_2}=-\frac{2}{k}\mathbf{1}\left[i_2\in\mathcal{I}\left[\boldsymbol{X}_{-i_1},X_{i_1};\mathcal{S},k\right]\right]$ and $V\left[\mathcal{S},k\right]_{i_1,i_2}=\frac{1}{k^2}\left|\{1\leq i\leq n:\{i_1,i_2\}\subseteq\mathcal{I}\left[\boldsymbol{X}_{-i},X_i;\mathcal{S},k\right]\}\right|$.

For a more efficient implementation we maintain $A\left[\mathcal{S}^{(t)}\right]$. Updating this matrix is easy and efficient since $A\left[\mathcal{S}\cup\{j\}\right]_{i_1,i_2}=A\left[\mathcal{S}\right]_{i_1,i_2}-x_{j,i_1}x_{j,i_2}$.

In each iteration, we compute the distances $\|X_{\mathcal{S}^{(t)}\cup\{j\},i_1}-X_{\mathcal{S}^{(t)}\cup\{j\},i_2}\|_2^2$ for all $j\notin\mathcal{S}^{(t)}$ and $1\leq i_1,i_2\leq n$. This computation is efficient since $\|X_{\mathcal{S}\cup\{j\},i_1}-X_{\mathcal{S}\cup\{j\},i_2}\|_2^2=\|X_{\mathcal{S},i_1}-X_{\mathcal{S},i_2}\|_2^2+\left(x_{j,i_1}-x_{j,i_2}\right)^2$, so computing the distance matrices and the $k_{\max}$ ranked nearest neighbors of each sample for each added probe in each iteration is done in $O\left(n^2m\right)$, and updating the distance matrix between iterations is done in $O\left(n^2\right)$.

Given the ranked nearest neighbors, we can set $U[\mathcal{S},0]=0$ and $V[\mathcal{S},0]=0$, and compute $U[\mathcal{S},k+1]$ from $U[\mathcal{S},k]$ in $O(n)$ and $V[\mathcal{S},k+1]$ from $V[\mathcal{S},k]$ in $O(nk)$.

Using these improvements, each iteration (minimizing over both $1 \le k \le k_{\max}$ and $1 \le j \le m$) can be done in $O\left(mn^2 + mnk_{\max}^2\right)$, and the overall complexity is $O\left(lmn\left(n+k_{\max}^2\right)\right)$. The complexity of imputation is $O\left(n|\mathcal{S}|+mk\right)$ (finding the k nearest neighbors and averaging them).

2.3 Locally weighted averaging

2.3.1 Selection and imputation

For a new observed test vector $Y_{\mathcal{S}}$, we compute its distance from all training samples and generate weights from these distances, which we use to impute a weighted average of the training samples. Let $\hat{\sigma}^2[\mathcal{S}] = \frac{1}{n(n-1)}\sum_{i_1 \ne i_2}\|X_{\mathcal{S},i_1} - X_{\mathcal{S},i_2}\|_2^2$, then the unnormalized weights are $\tilde{w}_i = \exp\left(-\frac{\|Y_{\mathcal{S}}-X_{\mathcal{S},i}\|_2^2}{2\alpha\hat{\sigma}^2[\mathcal{S}]}\right)$ and the actual weights are $w_i = \frac{\tilde{w}_i}{\sum_{i'=1}^{n}\tilde{w}_{i'}}$. α is a parameter of the method which scales the mean distance. The imputation is $Y = \sum_{i=1}^{n} w_i X_i$.

Selection is done by forward greedy selection, where each probe is chosen to minimize the LOOCV error. This error is

$$\mathcal{J}[\boldsymbol{X};\mathcal{S},\alpha] = \sum_{i=1}^{n}\sum_{j\notin\mathcal{S}}\left(x_{j,i} - \sum_{i'\ne i} w_{i'}x_{j,i}\right)^2$$

Initially, $\mathcal{S}^{(0)} = \phi$. Then, in each new iteration, all probe candidates $j \notin \mathcal{S}^{(t)}$ are examined to minimize the LOOCV error:

$$\hat{j}_{\mathcal{S}^{(t)}} = \arg\min_{j\notin\mathcal{S}^{(t)}} \mathcal{J}[\boldsymbol{X};\mathcal{S}\cup\{j\},\alpha]$$

Then, $\mathcal{S}^{(t+1)} = \mathcal{S}^{(t)} \cup \{\hat{j}_{\mathcal{S}^{(t)}}\}$. α can also be optimized (out of some predefined space Θ) to minimize the LOOCV error:

$$\langle\hat{j}_{\mathcal{S}^{(t)}}, \hat{\alpha}\rangle = \arg\min_{j\notin\mathcal{S}^{(t)},\alpha\in\Theta} \mathcal{J}[\boldsymbol{X};\mathcal{S}\cup\{j\},\alpha]$$

2.3.2 Implementation

In each iteration, we need to compute for each candidate probe the LOOCV error when it is added to the current probeset. To compute this error we first need to compute for each of the n samples the $n-1$ distances and corresponding weights. As before, computing the distances using the probeset $\mathcal{S}\cup\{j\}$ given the distances using $\mathcal{S}$ is done in $O\left(n^2\right)$. Hence, computing the normalized weights for all candidate probes and all samples only requires $O\left(mn^2\right)$ operations.

To compute the LOOCV error without generating the whole imputed vectors, we use:

$$\sum_{j\notin\mathcal{S}}\left(x_{j,i} - \sum_{i'\ne i} w_{i'}x_{j,i}\right)^2 = X_{-\mathcal{S},i}^T X_{-\mathcal{S},i} - 2\sum_{i'\ne i} w_{i'} X_{-\mathcal{S},i}^T X_{-\mathcal{S},i'} + \sum_{i_1,i_2\ne i} w_{i_1}w_{i_2} X_{-\mathcal{S},i_1}^T X_{-\mathcal{S},i_2}$$

As before we maintain the inner product matrix $A[\mathcal{S}]$. Computing the errors for all m candidate genes, then, requires $O\left(mn^3\right)$ operations. If we optimize also over α, then the overall complexity is $O\left(lm|\Theta|n^3\right)$.

For both kNN and locally weighted averaging, the main code is written in Python, but all the scores computations are written in C for faster running times.

2.3.3 Probe-specific kernel width

We also implemented a variant of the Locally Weighted Averaging method where each probe can have its own kernel width. In this case, the unnormalized weights are $\tilde{w}_i = \exp\left(-\frac{\sum_{j\in\mathcal{S}} \beta_j (y_j - x_{j,i})^2}{2\alpha\hat{\sigma}^2[\mathcal{S}]}\right)$. Due to the invariance of multiplying all β's and α by the same constant, we set the first kernel width to 1.

To choose the probes and the kernel widths, we initialize with the empty probeset $\mathcal{S}^{(0)} = \phi$ and then repeat for $t = 1, \ldots, l$:

1. Fix the current probeset $\mathcal{S}^{(t-1)}$ and the kernel widths $\beta^{(t-1)}$. Add a new probe j with $\beta_j = 1$ to $\mathcal{S}^{(t-1)}$ and choose new value $\alpha^{(t)}$ to minimize the LOOCV error for $\mathcal{S}^{(t-1)} \cup \{j\}$ (with the kernel widths $\beta^{(t-1)}$) and $\alpha^{(t)}$.
2. Fix $\alpha^{(t)}$ and the new probeset $\mathcal{S}^{(t)} = \mathcal{S}^{(t-1)} \cup \{j\}$.
3. Choose $\beta^{(t)}$ by looping over $j \in \mathcal{S}^{(t)}$ and choosing $\beta_j^{(t)}$ to minimize the LOOCV error with $\beta_{j'}^{(t)}$ fixed for all $j' \neq j$, until convergence.

2.3.4 Probabilistic interpretation

Let $Z_1, \ldots, Z_n \in \mathbb{N}$ be discrete random variable with no prior, and $X_i | Z_i = k \sim \mathcal{N}\left(\mu_k; \sigma^2 I\right)$. Since there is no prior on Z, optimizing the likelihood (using hard assignments) over the hidden variables Z and the parameters μ gives: $Z_i = i$ and $\mu_i = X_i$. This gives the generative model: $Z \sim U(1, n)$ and $X | Z = i \sim \mathcal{N}\left(X_i; \sigma^2 I\right)$. Using this generative model, we do the imputation by computing the expected value of the missing expression values:

$$\begin{aligned}
\mathbb{E}\left[Y_{-\mathcal{S}} | Y_{\mathcal{S}}\right] &= \sum_{i=1}^{n} P\left(Z = i | Y_{\mathcal{S}}\right) \mathbb{E}\left[Y_{-\mathcal{S}} | Z = i\right] \\
&= \sum_{i=1}^{n} \frac{P\left(Y_{\mathcal{S}} | Z = i\right)}{\sum_{i'=1}^{n} P\left(Y_{\mathcal{S}} | Z = i'\right)} X_{-\mathcal{S},i} \\
&= \sum_{i=1}^{n} X_{-\mathcal{S},i} \frac{\exp\left(-\frac{1}{2\sigma^2} \sum_{j\in\mathcal{S}} \left(y_j - x_{j,i}\right)^2\right)}{\sum_{i=1}^{n} \exp\left(-\frac{1}{2\sigma^2} \sum_{j\in\mathcal{S}} \left(y_j - x_{j,i'}\right)^2\right)} \\
&= \sum_{i=1}^{n} X_{-\mathcal{S},i} \frac{\tilde{w}_i}{\sum_{i'=1}^{n} \tilde{w}_{i'}} = \sum_{i=1}^{n} w_i X_{-\mathcal{S},i}
\end{aligned}$$

Hence, selection corresponds to learning a generative model using maximum likelihood estimation of both μ and hard assignments to Z, and imputation corresponds to computing the expected value under this model over the unobserved variables given the observed ones.

σ^2 cannot be estimated by maximum likelihood estimation since when it is not fixed, the likelihood is unbounded and goes to infinity as $\sigma^2 \to 0$: $\log P(\boldsymbol{X} | \boldsymbol{Z}; \mu, \sigma^2) = -\frac{mn}{2} \log(2\pi) - mn \log \sigma$. We can estimate σ^2 using the LOOCV marginal log-likelihood:

$$
\begin{aligned}
l(\boldsymbol{X};\mu,\sigma^2) &= \sum_{i=1}^{n}\log\left(\frac{1}{n-1}\sum_{i'\neq i}\left(2\pi\sigma^2\right)^{-m/2}\exp\left(-\frac{\|X_i-X_{i'}\|_2^2}{2\sigma^2}\right)\right) \\
&= -n\log(n-1)-\frac{mn}{2}\log\left(2\pi\sigma^2\right)+\sum_{i=1}^{n}\log\left(\sum_{i'\neq i}\exp\left(-\frac{\|X_i-X_{i'}\|_2^2}{2\sigma^2}\right)\right) \\
\frac{\partial l(\boldsymbol{X};\mu,\sigma^2)}{\partial\sigma^2} &= -\frac{mn}{2\sigma^2}+\sum_{i=1}^{n}\frac{\sum\limits_{i'\neq i}\exp\left(-\frac{\|X_i-X_{i'}\|_2^2}{2\sigma^2}\right)\frac{\|X_i-X_{i'}\|_2^2}{2\sigma^4}}{\sum\limits_{i'\neq i}\exp\left(-\frac{\|X_i-X_{i'}\|_2^2}{2\sigma^2}\right)} \\
\sigma^2 &= \frac{1}{mn}\sum_{i=1}^{n}\sum_{i'\neq i}w_{i,i'}\|X_i-X_{i'}\|_2^2
\end{aligned}
$$

where $w_{i,i'}=\exp\left(-\frac{\|X_i-X_{i'}\|_2^2}{2\sigma^2}\right)$ is itself a function of σ^2. This can be solved by iterative methods. In practice, this method of choosing σ^2 performs poorly due to the assumption of independence between different probes given the class variable Z (the Naive Bayes assumption), which is false in practice and leads to double-counting. In practice, we use $\frac{\alpha}{n(n-1)}\sum\limits_{i_1\neq i_2}\|X_{\mathcal{S},i_1}-X_{\mathcal{S},i_2}\|_2^2$ as detailed above.

2.4 Generative models

2.4.1 Imputation

Given a probability distribution $P(X)$ over expression vectors $X\in\mathbb{R}^m$ and a set of selected probes $\mathcal{S}$, we perform imputation by computing the expected value over the unobserved probes given the observed ones:

$$
\begin{aligned}
Y_{-\mathcal{S}} &= \int P(X_{-\mathcal{S}}|X_{\mathcal{S}})X_{-\mathcal{S}}dX_{-\mathcal{S}} \\
&= \frac{1}{P(X_{\mathcal{S}})}\int P(X_{-\mathcal{S}},X_{\mathcal{S}})X_{-\mathcal{S}}dX_{-\mathcal{S}}
\end{aligned}
$$

The choice of expected value as imputation method is done to minimize the squared error. We could also impute using the most likely assignment to the unobserved expression values:

$$
Y_{-\mathcal{S}}^{\mathrm{MAP}} = \arg\max_{X_{-\mathcal{S}}}P(X_{\mathcal{S}},X_{-\mathcal{S}})
$$

2.4.2 Selection

Now assume we have the generative model $P(X)$ but not the set of selected probes $\mathcal{S}$. We want to minimize the expected squared error of the imputation:

$$\begin{aligned}\epsilon[\mathcal{S}] &= \int P(X')\|X'_{-\mathcal{S}} - \mathbb{E}[X_{-\mathcal{S}}|X'_{\mathcal{S}}]\|_2^2 dX' \\ &= \mathbb{E}\left[\|X'_{-\mathcal{S}} - \mathbb{E}[X_{-\mathcal{S}}|X'_{\mathcal{S}}]\|_2^2\right] = \mathbb{E}\left[\sum_{j\notin\mathcal{S}}\left(X'_j - \mathbb{E}[X_j|X'_{\mathcal{S}}]\right)^2\right] \\ &= \sum_{j\notin\mathcal{S}}\mathbb{E}\left[\left(X'_j - \mathbb{E}[X_j|X'_{\mathcal{S}}]\right)^2\right] = \sum_{j\notin\mathcal{S}}\mathrm{Var}[X_j|X_{\mathcal{S}}]\end{aligned}$$

This shows the intuitive notion that the expected error for a set of probes is simply the remaining variance conditioned on these probes being observed. The problem of finding the set $\mathcal{S}$ of some fixed size that minimizes $\epsilon[\mathcal{S}]$ is generally hard. In practice, if we can compute $\mathrm{Var}[X_j|X_{\mathcal{S}}]$ for any $\mathcal{S}$, we can use a greedy algorithm: initialize $\mathcal{S}^{(0)} = \phi$, then for $t = 1, 2, \ldots, l$: compute $v_j^{(t)} = \sum_{j'\notin\mathcal{S}\cup\{j\}} \mathrm{Var}\left[X_{j'}|X_{\mathcal{S}\cup\{j\}}\right]$ for $j \notin \mathcal{S}^{(t-1)}$ and let $\mathcal{S}^{(t)} = \mathcal{S}^{(t-1)} \cup \{\arg\min_j v_j^{(t)}\}$.

2.4.3 Learning

We usually don't have a single generative model $P(X)$ as used above. Suppose we have a family of models $\mathcal{P}$ with a probability measure Q over $\mathcal{P}$ and a training set $\mathcal{D}$. Then, given a set of probes $\mathcal{S}$, we perform imputation by: $Y_{-\mathcal{S}} = \mathbb{E}_{P\sim Q(P|\mathcal{D})}\left[\mathbb{E}_{Y'_{-\mathcal{S}}\sim P(X_{-\mathcal{S}}|X_{\mathcal{S}})}\left[Y'_{-\mathcal{S}}|Y_{\mathcal{S}}\right]\right]$. This is a Bayesian Model Averaging approach where we integrate over the models based on their posterior probability given the training data.

To select a probeset $\mathcal{S}$, we want to minimize the squared error as before, and choose:

$$\hat{\mathcal{S}} = \arg\min_{\mathcal{S}\subseteq\{1,2,\ldots,m\}:|\mathcal{S}|=l} \sum_{j\notin\mathcal{S}} \mathbb{E}_{P(X)\sim Q(P|\mathcal{D})}\left[\mathrm{Var}_{X\sim P(X)}[X_j|X_{\mathcal{S}}]\right]$$

Instead of computing the expectation over all models, we can approximate it by using only the most likely model $\arg\max_{P\in\mathcal{P}} Q(P|\mathcal{D})$ or by samples, if we can sample from $Q(P|\mathcal{D})$.

2.5 Gaussian models

Suppose $X \sim \mathcal{N}(\mu; C)$. Then for a probeset $\mathcal{S}$,

$$X_{-\mathcal{S}}|X_{\mathcal{S}} \sim \mathcal{N}\left(\mu_{-\mathcal{S}} + C_{-\mathcal{S},\mathcal{S}}C_{\mathcal{S},\mathcal{S}}^{-1}(X_{\mathcal{S}} - \mu_{\mathcal{S}}); C_{-\mathcal{S},-\mathcal{S}} - C_{-\mathcal{S},\mathcal{S}}C_{\mathcal{S},\mathcal{S}}^{-1}C_{\mathcal{S},-\mathcal{S}}\right)$$

and the expected squared error is:

$$\mathcal{J}[\mathcal{S}] = \mathrm{Tr}\left[C_{-\mathcal{S},-\mathcal{S}} - C_{-\mathcal{S},\mathcal{S}}C_{\mathcal{S},\mathcal{S}}^{-1}C_{\mathcal{S},-\mathcal{S}}\right]$$

For any symmetric invertible matrix $\begin{pmatrix} A & v \\ v^T & b \end{pmatrix}$, it holds that

$$\begin{pmatrix} A & v \\ v^T & b \end{pmatrix}^{-1} = \begin{pmatrix} \frac{A^{-1}vv^TA^{-1}}{b-v^TA^{-1}v} + A^{-1} & \frac{A^{-1}v}{v^TA^{-1}v-b} \\ \frac{v^TA^{-1}}{v^TA^{-1}v-b} & \frac{1}{b-v^TA^{-1}v} \end{pmatrix}$$

Using this property, we can efficiently invert $C_{\mathcal{S}\cup\{j\},\mathcal{S}\cup\{j\}}$ given that we have already computed $C_{\mathcal{S},\mathcal{S}}^{-1}$:

$$C^{-1}_{\mathcal{S}\cup\{j\},\mathcal{S}\cup\{j\}} = \begin{pmatrix} \frac{C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j} C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}}}{C_{j,j} - C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j}} + C^{-1}_{\mathcal{S},\mathcal{S}} & \frac{C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j}}{C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j} - C_{j,j}} \\ \frac{C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}}}{C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j} - C_{j,j}} & \frac{1}{C_{j,j} - C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j}} \end{pmatrix}$$

Now we can compute the difference $\Delta[\mathcal{S}]_j = \mathcal{J}[\mathcal{S}\cup\{j\}] - \mathcal{J}[\mathcal{S}]$ in expected errors due to the addition of a single probe j:

$$\Delta[\mathcal{S}]_j = -C_{j,j} + \mathrm{Tr}\left[C_{-\mathcal{S},\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},-\mathcal{S}}\right] - \mathrm{Tr}\left[C_{-\mathcal{S}-\{j\},\mathcal{S}\cup\{j\}} C^{-1}_{\mathcal{S}\cup\{j\},\mathcal{S}\cup\{j\}} C_{\mathcal{S}\cup\{j\},-\mathcal{S}-\{j\}}\right] \quad (1)$$

Now let $\Sigma[\mathcal{S}] = C_{-\mathcal{S},\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},-\mathcal{S}}$ and $\Sigma[\mathcal{S}]_j = C_{-\mathcal{S}-\{j\},\mathcal{S}\cup\{j\}} C^{-1}_{\mathcal{S}\cup\{j\},\mathcal{S}\cup\{j\}} C_{\mathcal{S}\cup\{j\},-\mathcal{S}-\{j\}}$, and let r denote the i'th index in $-\mathcal{S}-\{j\}$, then:

$$\begin{aligned}
(\Sigma[\mathcal{S}]_j)_{i,i} &= \left(C_{-\mathcal{S}-\{j\},\mathcal{S}\cup\{j\}} C^{-1}_{\mathcal{S}\cup\{j\},\mathcal{S}\cup\{j\}} C_{\mathcal{S}\cup\{j\},-\mathcal{S}-\{j\}}\right)_{i,i} \\
&= \sum_{k=1}^{t+1} \left(C_{-\mathcal{S}-\{j\},\mathcal{S}\cup\{j\}}\right)_{i,k} \left(C^{-1}_{\mathcal{S}\cup\{j\},\mathcal{S}\cup\{j\}} C_{\mathcal{S}\cup\{j\},-\mathcal{S}-\{j\}}\right)_{k,i} \\
&= C_{r,j} \left(C^{-1}_{\mathcal{S}\cup\{j\},\mathcal{S}\cup\{j\}} C_{\mathcal{S}\cup\{j\},-\mathcal{S}-\{j\}}\right)_{t+1,i} + \sum_{k=1}^{t} (C_{-\mathcal{S},\mathcal{S}})_{i,k} \left(C^{-1}_{\mathcal{S}\cup\{j\},\mathcal{S}\cup\{j\}} C_{\mathcal{S}\cup\{j\},-\mathcal{S}-\{j\}}\right)_{k,i} \\
&= C_{r,j} \left(\frac{C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},r}}{C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j} - C_{j,j}} + \frac{C_{j,r}}{C_{j,j} - C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j}}\right) + \\
&+ C_{r,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} \left(\left(\frac{C_{\mathcal{S},j} C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}}}{C_{j,j} - C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j}} + I\right) C_{\mathcal{S},r} + \frac{C_{\mathcal{S},j} C_{j,r}}{C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j} - C_{j,j}}\right) \\
&= (\Sigma[\mathcal{S}])_{i,i} + \frac{\sigma[i;j|\mathcal{S}]^2}{C_{j,j} - C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j}} \\
&= (\Sigma[\mathcal{S}])_{i,i} + \frac{\sigma[i;j|\mathcal{S}]^2}{\sigma[j;j|\mathcal{S}]} \\
\sigma[i;j|\mathcal{S}] &= C_{i,j} - C_{i,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j}
\end{aligned}$$

Plugging this result into Equation 1 gives:

$$\Delta[\mathcal{S}]_j = -\sum_{i\notin\mathcal{S}\cup\{j\}} \frac{\sigma[i;j|\mathcal{S}]^2}{\sigma[j;j|\mathcal{S}]} - C_{j,j} + C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j} = -\sum_{i\notin S} \frac{\sigma[i;j|\mathcal{S}]^2}{\sigma[j;j|\mathcal{S}]}$$

2.5.1 Principal Component Analysis

PCA is a specific case of a Gaussian model, where a low-rank approximation of the covariance matrix is used. Let $\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$ and $\hat{\Sigma} = \frac{1}{n-1}\sum_{i=1}^{n} \left(X_i - \bar{X}\right)\left(X_i - \bar{X}\right)^T$. The eigen-decomposition of the empirical covariance matrix is $\hat{\Sigma} = U\Lambda U^T$, where Λ is diagonal and $U \in \mathbb{R}^{m\times m}$ satisfies $U^T U = I$. The total variance is $\sum_{i=1}^{m} \Lambda_{i,i}$. The i'th principal component U_i explains $\frac{\Lambda_{i,i}}{\mathrm{Tr}[\Lambda]}$ of the variance. By using only the first k principal components we have a rank-k approximation to $\hat{\Sigma}$ which explains $\frac{\sum_{i=1}^{k} \Lambda_{i,i}}{\mathrm{Tr}[\Lambda]}$ of the variance in the data. Denote by $U_{1:k} \in \mathbb{R}^{m\times k}$ the matrix obtained by

taking the first k columns of U, and $\Lambda_{1:k}$ the diagonal matrix obtained by using the first k elements on the diagonal of Λ. Then $U_{1:k}\Lambda_{1:k}U_{1:k}^T$ is a rank-k approximation to $\hat{\Sigma}$. Since this approximation is singular and uses no components from the space orthogonal to $U_{1:k}$, which contains $\frac{\sum_{i>k}\Lambda_{i,i}}{\mathrm{Tr}[\Lambda]}$ of the variance, we add back this space with a total contribution to the variance as it had originally by adding $\frac{1}{m-k}\left(\sum_{i>k}\Lambda_{i,i}\right)\left(I-U_{1:k}U_{1:k}^T\right)$. Let $\Sigma = U_{1:k}\Lambda_{1:k}U_{1:k}^T + \frac{1}{m-k}\left(\sum_{i>k}\Lambda_{i,i}\right)\left(I-U_{1:k}U_{1:k}^T\right)$, then we model the data as $X \sim \mathcal{N}(\mu;\Sigma)$ where $\mu = \bar{X}$.

To choose the number of principal components to be used, we use leave-one-out cross-validation and compute the total log-likelihood of the left-out samples given the covariance estimated on the remaining ones. This is done efficiently by not computing the full covariance matrix or its inverse at any stage, but rather only the results of their product with the test sample.

2.5.2 PCA Regression

In addition to using PCA as a specific case of a Gaussian model, we can use PCA regression directly for probe selection and imputation. For a probeset $\mathcal{S}$, let $v = \left(U_{\mathcal{S},1:k}^T U_{\mathcal{S},1:k}\right)^{-1} U_{\mathcal{S},1:k}^T \left(Y_{\mathcal{S}} - \mu_{\mathcal{S}}\right)$ be the least-squares solution for the PCA coefficients, and $U_{-\mathcal{S},1:k}v + \mu_{-\mathcal{S}}$ is the imputation. The reconstruction squared error is $\left\| U_{-\mathcal{S},1:k}\left(U_{\mathcal{S},1:k}^T U_{\mathcal{S},1:k}\right)^{-1} U_{\mathcal{S},1:k}^T \left(Y_{\mathcal{S}} - \mu_{\mathcal{S}}\right) - \left(Y_{-\mathcal{S}} - \mu_{-\mathcal{S}}\right)\right\|_2^2$.

We use forward greedy selection to choose probes to minimize the LOOCV reconstruction error. Since $U_{\mathcal{S}\cup\{j\},1:k}^T U_{\mathcal{S}\cup\{j\},1:k} = U_{\mathcal{S},1:k}^T U_{\mathcal{S},1:k} + U_{j,1:k}^T U_{j,1:k}$, we can compute the inverse easily. Let $A_{\mathcal{S}} = \left(U_{\mathcal{S},1:k}^T U_{\mathcal{S},1:k}\right)^{-1}$ and $u_{\mathcal{S}} = A_{\mathcal{S}}U_{j,1:k}^T$, then

$$\begin{aligned} A_{\mathcal{S}\cup\{j\}} &= \left(U_{\mathcal{S}\cup\{j\},1:k}^T U_{\mathcal{S}\cup\{j\},1:k}\right)^{-1} \\ &= A_{\mathcal{S}} - \frac{A_{\mathcal{S}}U_{j,1:k}^T U_{j,1:k} A_{\mathcal{S}}}{1 + U_{j,1:k}A_{\mathcal{S}}U_{j,1:k}^T} \\ &= A_{\mathcal{S}} - \frac{u_{\mathcal{S}}u_{\mathcal{S}}^T}{1 + U_{j,1:k}u_{\mathcal{S}}} \end{aligned}$$

Given the current set $\mathcal{S}$, we compute the LOOCV error using $\mathcal{S}\cup\{j\}$ for all $j \notin \mathcal{S}$, $1 \le k \le \min[n-2, m]$ and $\lambda \in \{2^x | x \in \{\alpha_{\min}, \ldots, \alpha_{\max}\}\}$ where in our experiments $\alpha_{\min} = -20, \alpha_{\max} = 0$.

2.5.3 Shrinkage covariance estimation

Rather than using the maximum likelihood covariance estimate $\frac{1}{n}\sum_{i=1}^{n}\left(X_i - \bar{X}\right)\left(X_i - \bar{X}\right)^T$, we use this estimate for the main diagonal but shrink the off-diagonal elements by α, so: $\hat{\Sigma} = \frac{\alpha}{n}\sum_{i=1}^{n}\left(X_i - \bar{X}\right)\left(X_i - \bar{X}\right)^T + (1-\alpha)D$ where D is a diagonal matrix with $D_{j,j} = \frac{1}{n}\sum_{i=1}^{n}\left(X_{j,i} - \bar{X}_j\right)^2$.

We choose α to maximize the LOOCV likelihood. Let C be the estimated covariance matrix using the full training data: $C = \frac{1}{n}\sum_{i=1}^{n}X_i X_i^T - \left(\frac{1}{n}\sum_{i=1}^{n}X_i\right)\left(\frac{1}{n}\sum_{i=1}^{n}X_i\right)^T$. The estimated covariance matrix C_{-i} when leaving out sample i is $C_{-i} = \frac{1}{n-1}\sum_{i'\neq i}X_i X_i^T - \left(\frac{1}{n-1}\sum_{i'\neq i}X_{i'}\right)\left(\frac{1}{n-1}\sum_{i'\neq i}X_{i'}\right)^T$, and D_{-i} the diagonal matrix defined by $\left(D_{-i}\right)_{j,j} = \frac{1}{n-1}\sum_{i'\neq i}\left(X_{j,i} - \left(\mu_{-i}\right)_j\right)^2$, where $\mu_{-i} = \frac{1}{n-1}\sum_{i'\neq i}X_{i'}$.

The LOOCV likelihood is:

$$-2l\left(\alpha:\mathcal{D}\right)=nm\log\left(2\pi\right)+\sum_{i=1}^{n}\log\left|\alpha C_{-i}+(1-\alpha)D_{-i}\right|+\sum_{i=1}^{n}\left(X_i-\mu_{-i}\right)^T\left(\alpha C_{-i}+\left(1-\alpha\right)D_{-i}\right)^{-1}\left(X_i-\mu_{-i}\right) \tag{2}$$

An analytical solution for α based on the LOOCV likelihood is hard, and we compare two approaches:

1. A numerical search to minimize Equation 2. Let $M_{-i}=D_{-i}^{-1/2}\left(C_{-i}-D_{-i}\right)D_{-i}^{-1/2}$ and let $M_{-i}=U_{-i}\Lambda_{-i}U_{-i}^T$ be its eigen-decomposition. Also let $d_{-i,j}=\left(D_{-i}\right)_{j,j}$ and $\lambda_{-i,j}=\left(\Lambda_{-i}\right)_{j,j}$. Then

$$\begin{aligned}
\alpha C_{-i}+\left(1-\alpha\right)D_{-i} &= D_{-i}+\alpha\left(C_{-i}-D_{-i}\right)\\
&= D_{-i}^{1/2}\left(I+\alpha D_{-i}^{-1/2}\left(C_{-i}-D_{-i}\right)D_{-i}^{-1/2}\right)D_{-i}^{1/2}\\
&= D_{-i}^{1/2}\left(I+\alpha U_{-i}\Lambda_{-i}U_{-i}^T\right)D_{-i}^{1/2}\\
&= D_{-i}^{1/2}U_{-i}\left(\alpha\Lambda_{-i}+I\right)U_{-i}^TD_{-i}^{1/2}\\
\log\left|\alpha C_{-i}+\left(1-\alpha\right)D_{-i}\right| &= \sum_{j=1}^{m}\left(\log\left(d_{-i,j}\right)+\log\left(1+\alpha\lambda_{-i,j}\right)\right)\\
\left(\alpha C_{-i}+\left(1-\alpha\right)D_{-i}\right)^{-1} &= D_{-i}^{-1/2}U_{-i}\left(\alpha\Lambda_{-i}+I\right)^{-1}U_{-i}^TD_{-i}^{-1/2}\\
-2l\left(\alpha:\mathcal{D}\right) &= nm\log\left(2\pi\right)+\sum_{i=1}^{n}\sum_{j=1}^{m}\left(\log\left(d_{-i,j}\right)+\log\left(1+\alpha\lambda_{-i,j}\right)\right)+\\
&+ \sum_{i=1}^{n}\left(X_i-\mu_{-i}\right)^TD_{-i}^{-1/2}U_{-i}\left(\alpha\Lambda_{-i}+I\right)^{-1}U_{-i}^TD_{-i}^{-1/2}\left(X_i-\mu_{-i}\right)\\
&= nm\log\left(2\pi\right)+\sum_{i=1}^{n}\sum_{j=1}^{m}\left(\log\left(d_{-i,j}\right)+\log\left(1+\alpha\lambda_{-i,j}\right)\right)+\\
&+ \sum_{i=1}^{n}\sum_{j=1}^{m}\frac{v_{-i,j}^2}{1+\alpha\lambda_{-i,j}}
\end{aligned}$$

   where $V_{-i}=U_{-i}^TD_{-i}^{-1/2}\left(X_i-\mu_{-i}\right)$.

2. Choose α as previously suggested [3]. $\alpha^*=\frac{\sum_{i\neq j}\widehat{\mathrm{Var}}(s_{i,j})}{\sum_{i\neq j}s_{i,j}^2}$ where s_{j_1,j_2} is the unbiased empirical estimate of the covariance between probes j_1 and j_2: $s_{j_1,j_2}=\frac{1}{n-1}\sum_{i=1}^{n}\left(x_{j_1,i}-\bar{x}_{j_1}\right)\left(x_{j_2,i}-\bar{x}_{j_2}\right)$ and $\widehat{\mathrm{Var}}\left(s_{i,j}\right)$ is the empirical unbiased estimate of the variance of $s_{i,j}$.

In practice, both methods perform quite similarly, with the actual α values chosen very similar as well.

2.5.4 Regularized covariance estimation

In this formulation, we place a prior on the estimated covariance matrix $\hat{\Sigma}$, which is a Gaussian prior on the entries of the precision matrix Σ^{-1}: $P\left(\Sigma\right)=\left(2\pi\right)^{\frac{-m^2}{2}}\sigma^{-m^2}\exp\left(\frac{-1}{2\sigma^2}\sum_{i=1}^{m}\sum_{j=1}^{m}\left(\Sigma_{i,j}^{-1}\right)^2\right)$. Then, the likelihood is:

$$
\begin{aligned}
-2l\left(\Sigma : \boldsymbol{X}\right) &= mn\log\left(2\pi\right) - n\log\left|\Sigma^{-1}\right| + \sum_{i=1}^{n}\left(X_i - \bar{X}\right)^T\Sigma^{-1}\left(X_i - \bar{X}\right) + \\
&+ m\log\left(2\pi\right) + 2m^2\log\sigma + \frac{1}{\sigma^2}\operatorname{Tr}\left[\Sigma^{-2}\right] \\
\frac{\partial\left(-2l\left(\Sigma : \boldsymbol{X}\right)\right)}{\partial\Sigma^{-1}} &= -n\Sigma + \sum_{i=1}^{n}\left(X_i - \bar{X}\right)\left(X_i - \bar{X}\right)^T + \frac{2}{\sigma^2}\Sigma^{-1}
\end{aligned}
$$

Let $\frac{1}{n}\sum_{i=1}^{n}\left(X_i - \bar{X}\right)\left(X_i - \bar{X}\right)^T = U\Lambda U^T$ be the eigen-decomposition of the empirical covariance matrix. Setting the gradient to 0:

$$
\begin{aligned}
0 &= \Sigma^2 - \frac{\Sigma}{n}\sum_{i=1}^{n}\left(X_i - \bar{X}\right)\left(X_i - \bar{X}\right)^T - \frac{2}{n\sigma^2}I \\
\Sigma &= \frac{1}{2n}\sum_{i=1}^{n}\left(X_i - \bar{X}\right)\left(X_i - \bar{X}\right)^T \pm \left(\frac{1}{4n^2}\left(\sum_{i=1}^{n}\left(X_i - \bar{X}\right)\left(X_i - \bar{X}\right)^T\right)^2 + \frac{2}{n\sigma^2}I\right)^{1/2} \\
&= \frac{1}{2}U\Lambda U^T \pm \left(\frac{1}{4}U\left(\Lambda^2 + \frac{8}{n\sigma^2}I\right)U^T\right)^{1/2} \\
&= U\left(\frac{1}{2}\Lambda + \frac{1}{2}\sqrt{\Lambda^2 + \frac{8}{n\sigma^2}I}\right)U^T
\end{aligned}
$$

Since $\frac{8}{n\sigma^2} > 0$, the minus solution results in a matrix which is not positive semi-definite, so the only solution is the plus solution.

2.5.5 L_2 regularized inverse covariance estimation with different regularization coefficients

The analytical solution to the L_2 regularized inverse covariance estimation described in Section 2.5.4 assumed a single regularization coefficient σ^2. Now we generalize this to the following optimization problem:

$$
\text{Maximize}_{\Sigma}\sum_{i=1}^{n}\left(-\frac{m}{2}\log\left(2\pi\right) - \frac{1}{2}\log\left|\Sigma\right| - \frac{1}{2}\left(X_i - \mu\right)^T\Sigma^{-1}\left(X_i - \mu\right)\right) - \sum_{j_1=1}^{m}\sum_{j_2=1}^{m}\lambda_{j_1,j_2}\left(\Sigma^{-1}\right)_{j_1,j_2}^2 \tag{3}
$$

Let $J = \Sigma^{-1}$ and $\hat{\Sigma} = \frac{1}{n}\sum_{i=1}^{n}\left(X_i - \mu\right)\left(X_i - \mu\right)^T$. The Equation 3 is equivalent to:

$$
\text{Maximize}_{\Sigma}\, n\log\left|J\right| - n\operatorname{Tr}\left[\hat{\Sigma}J\right] - \sum_{j_1=1}^{m}\sum_{j_2=1}^{m}\lambda_{j_1,j_2}J_{j_1,j_2}^2 \tag{4}
$$

Introduce Z with the constraint $Z = J$ and the Lagrange multipliers W, then the Lagrangian is:

$$
\begin{aligned}
\mathcal{L}(J,Z,W) &= n\,\mathrm{Tr}\left[\hat{\Sigma}J\right] - n\log|J| + \sum_{j_1=1}^{m}\sum_{j_2=1}^{m}\lambda_{j_1,j_2}Z_{j_1,j_2}^2 + \mathrm{Tr}\left[W(J-Z)\right] \\
\inf_{Z_{j_1,j_2}} \mathcal{L} &= \inf_{Z_{j_1,j_2}} \lambda_{j_1,j_2}Z_{j_1,j_2}^2 - W_{j_1,j_2}Z_{j_1,j_2} \\
Z_{j_1,j_2} &= \frac{W_{j_1,j_2}}{2\lambda_{j_1,j_2}} \\
\inf_{Z_{j_1,j_2}} \mathcal{L} &= \begin{cases} -\frac{W_{j_1,j_2}^2}{4\lambda_{j_1,j_2}} & \text{if } \lambda_{j_1,j_2} > 0 \\ 0 & \text{if } \lambda_{j_1,j_2} = 0 \text{ and } W_{j_1,j_2} = 0 \\ -\infty & \text{if } \lambda_{j_1,j_2} = 0 \text{ and } W_{j_1,j_2} \neq 0 \end{cases} \\
\inf_J \mathcal{L} &= \inf_J n\,\mathrm{Tr}\left[\left(\hat{\Sigma} + \frac{1}{n}W\right)J\right] - n\log|J| \\
\hat{\Sigma} + \frac{1}{n}W &= J^{-1} \\
\Sigma &= \hat{\Sigma} + \frac{1}{n}W \\
J &= \left(\hat{\Sigma} + \frac{1}{n}W\right)^{-1} \\
\inf_J \mathcal{L} &= n\left(m - \log|J|\right)
\end{aligned}
$$

Combining the above, we get the dual:

$$
\begin{aligned}
g(W) &= \inf_{J,Z} \mathcal{L}(J,Z,W) \\
&= \begin{cases} -\infty & \text{if } \exists j_1, j_2 : \lambda_{j_1,j_2} = 0 \text{ and } W_{j_1,j_2} \neq 0 \\ -n\left(\log|J| - m\right) - \sum_{j_1,j_2:\lambda_{j_1,j_2}>0} \frac{W_{j_1,j_2}^2}{4\lambda_{j_1,j_2}} & \text{otherwise} \end{cases}
\end{aligned}
$$

The dual objective is:

$$
\begin{aligned}
\text{Maximize}_W \quad & -n\log|J| - \sum_{j_1,j_2:\lambda_{j_1,j_2}>0} \frac{W_{j_1,j_2}^2}{4\lambda_{j_1,j_2}} \\
\text{s.t.} \quad & \forall j_1, j_2 : \lambda_{j_1,j_2} = 0 \Rightarrow W_{j_1,j_2} = 0
\end{aligned}
$$

For a dual feasible point W, the corresponding primal point is $J = \left(\hat{\Sigma} + \frac{1}{n}W\right)^{-1}$, the primal objective value is $n\,\mathrm{Tr}\left[\hat{\Sigma}J\right] - n\log|J| + \sum_{j_1=1}^{m}\sum_{j_2=1}^{m}\lambda_{j_1,j_2}J_{j_1,j_2}^2$, the dual objective value is $nm - n\log|J| - \sum_{j_1,j_2:\lambda_{j_1,j_2}>0} \frac{W_{j_1,j_2}^2}{4\lambda_{j_1,j_2}}$ and the duality gap is

$$
\eta(W) = n\,\mathrm{Tr}\left[\hat{\Sigma}J\right] + \sum_{j_1=1}^{m}\sum_{j_2=1}^{m}\lambda_{j_1,j_2}J_{j_1,j_2}^2 + \sum_{j_1,j_2:\lambda_{j_1,j_2}>0} \frac{W_{j_1,j_2}^2}{4\lambda_{j_1,j_2}} - nm \tag{5}
$$

We solve the dual by gradient ascent. Initialize $\Sigma^{(0)} = \alpha\hat{\Sigma} + (1-\alpha)D$ where D is a diagonal matrix with $D_{i,i} = \hat{\Sigma}_{i,i}$ for some α close to 1. Let $W^{(0)} = n\left(\Sigma^{(0)} - \hat{\Sigma}\right) = n(1-\alpha)\left(D - \hat{\Sigma}\right)$. Then $\Sigma^{(0)}$ is positive definite and $W^{(0)}$ is a feasible point. Now, for a feasible point W let $\Sigma = \hat{\Sigma} + \frac{1}{n}W$ and $J = \Sigma^{-1}$. The gradient is:

$$
\begin{aligned}
\nabla_W g &= n\nabla_W \log\left|\hat{\Sigma}+\frac{1}{n}W\right| - \frac{W_{j_1,j_2}}{2\lambda_{j_1,j_2}} \\
&= n\,() \\
\frac{\partial g}{\partial W_{j_1,j_2}} &= \begin{cases} J_{j_1,j_2} - \frac{W_{j_1,j_2}}{2\lambda_{j_1,j_2}} & \text{if } \lambda_{j_1,j_2} > 0 \\ J_{j_1,j_2} & \text{otherwise} \end{cases}
\end{aligned}
$$

Each step consists of computing the direction of change (the gradient G) and a line search to optimize t, the size of the move in that direction. For the current value of W, let $\Sigma = \hat{\Sigma} + \frac{1}{n}W$, $J = \Sigma^{-1} = \left(\hat{\Sigma} + \frac{1}{n}W\right)^{-1}$, $G = \nabla_W g$ and $M = \frac{1}{n}J^{1/2}GJ^{1/2}$. Also let $\psi_1, \ldots, \psi_n$ be the eigenvalues of M. Then:

$$
\begin{aligned}
f(t) &= g(W+tG) = n\log\left|\hat{\Sigma}+\frac{1}{n}W+\frac{t}{n}G\right| - \sum_{j_1,j_2:\lambda_{j_1,j_2}>0}\frac{\left(W_{j_1,j_2}+tG_{j_1,j_2}\right)^2}{4\lambda_{j_1,j_2}} \\
\hat{\Sigma}+\frac{1}{n}W+\frac{t}{n}G &= \Sigma+\frac{t}{n}G = \Sigma^{1/2}\left(I+\frac{t}{n}\Sigma^{-1/2}G\Sigma^{-1/2}\right)\Sigma^{1/2} \\
&= \Sigma^{1/2}\left(I+tM\right)\Sigma^{1/2} \\
\log\left|\hat{\Sigma}+\frac{1}{n}W+\frac{t}{n}G\right| &= \log|\Sigma| + \sum_i \log\left(1+t\psi_i\right)
\end{aligned}
$$

Note that if $\min_i (1+t\psi_i) > 0$ then $I+tM$ is positive definite and since Σ is positive definite (and therefore $\Sigma^{1/2}$ is positive definite), the product $\Sigma^{1/2}\left(I+tM\right)\Sigma^{1/2}$ is positive definite and it is equal to $\hat{\Sigma}+\frac{1}{n}W+\frac{t}{n}G$. Conversely, if $\hat{\Sigma}+\frac{1}{n}W+\frac{t}{n}G$ is positive definite then so is $\Sigma^{-1/2}\left(\hat{\Sigma}+\frac{1}{n}W+\frac{t}{n}G\right)\Sigma^{-1/2}$, which is equal to $I+tM$, and this implies that $1+t\psi_i > 0$ for all i. Hence, $1+t\psi_i > 0$ for all i is a necessary and sufficient condition for $\hat{\Sigma}+\frac{1}{n}W+\frac{t}{n}G$ being positive definite, and we constrain the line search to the range of values of t such that $1+t\psi_i > 0$ for all i. To find the optimal t within that range, we use Newton's method:

$$
\begin{aligned}
f'(t) &= n\sum_i \frac{\psi_i}{1+t\psi_i} - \sum_{j_1,j_2:\lambda_{j_1,j_2}>0}\frac{G_{j_1,j_2}\left(W_{j_1,j_2}+tG_{j_1,j_2}\right)}{2\lambda_{j_1,j_2}} \\
f''(t) &= -n\sum_i \frac{\psi_i^2}{\left(1+t\psi_i\right)^2} - \sum_{j_1,j_2:\lambda_{j_1,j_2}>0}\frac{G_{j_1,j_2}^2}{2\lambda_{j_1,j_2}}
\end{aligned}
$$

Each iteration of the gradient ascent procedure consists of:

- Computing $\Sigma = \hat{\Sigma} + \frac{1}{n}W$. $O(m^2)$.
- Computing the eigen-decomposition of Σ. $O(m^3)$.
- Computing $J = \Sigma^{-1}$. $O(m^3)$.
- Computing $J^{1/2}$ (using the eigen-decomposition already computed). $O(m^3)$.
- Computing the gradient G as described above. $O(m^2)$.

- Computing $M = J^{1/2}GJ^{1/2}$ and its eigen-decomposition. $O(m^3)$.
- Line search to find optimal t. $O(rm^2)$ where r is the number of inner line search iterations.
- Perform the move: $W \leftarrow W + tG$. $O(m^2)$.

Overall, an iteration takes $O(m^3)$ operations. Convergence is determined by comparing the duality gap to a threshold.

2.5.6 L_1 regularized inverse covariance estimation

We use the projected subgradient method [1] to estimate a (sparse) L_1 regularized inverse covariance matrix, and use this matrix as described above.

2.5.7 L_∞-regularized linear regression

Since our goal is to predict all probes from a small set of selected probes, a good linear model to use is one which automatically results in a small set of probes which have nonzero weights for predicting any of the other probes. If $W \in \mathbb{R}^{m\times m}$ is the weight matrix such $Y = W^T X$, we would like W to have few nonzero rows.

Additionally, we would like to control sparsity and shrinkage separately: controlling sparsity allows us to choose the number of probes that we will use, and a separate control of shrinkage allows us to use cross-validation to optimize the prediction accuracy separately from the resulting number of probes. Finally, we would like our optimization goal to be convex in the weights, for computational reasons. This prevents us from using something like an L_0 norm which only considers the number of nonzero rows.

One possible alternative is to use a sum of L_∞ norms: $\sum_{j=1}^{m} \|W_j\|_\infty = \sum_{j_1=1}^{m} \max_{j_2=1}^{m} |w_{j_1,j_2}|$. The advantages of this penalty term are that it is convex and can be optimized using coordinate descent, and it results in a solution which is sparse in nonzero rows, but at the same time its shrinking effects on the weights are minimal.

To solve for a single weight while keeping all others fixed, we have the following optimization problem:

$$\text{Minimize}_{w_{j_1,j_2}} \quad \frac{1}{2}\sum_{i=1}^{n}\left(\left(x_{j_2,i} - \sum_{j\neq j_1} w_{j,j_2}x_{j,i}\right) - w_{j_1,j_2}x_{j_1,i}\right)^2 + \lambda \max_{j=1}^{m} |w_{j_1,j}|$$

The partial derivative of the likelihood without the penalty term with respect to w_{j_1,j_2} is:

$$\frac{\partial L(W)}{\partial w_{j_1,j_2}} = \sum_{i=1}^{n} x_{j_1,i}\left(\sum_{j=1}^{m} w_{j,j_2}x_{j,i} - x_{j_2,i}\right)$$

w_{j_1,j_2} is optimal (keeping all other weights fixed) if $|w_{j_1,j_2}| < \max_{j=1}^{m} |w_{j_1,j}|$ and $\sum_{i=1}^{n} x_{j_1,i}\left(\sum_{j=1}^{m} w_{j,j_2}x_{j,i} - x_{j_2,i}\right) = 0$, or $|w_{j_1,j_2}| = \max_{j=1}^{m} |w_{j_1,j}|$ and $\sum_{i=1}^{n} x_{j_1,i}\left(\sum_{j=1}^{m} w_{j,j_2}x_{j,i} - x_{j_2,i}\right) > -\lambda$ (for $w_{j_1,j_2} > 0$), $\sum_{i=1}^{n} x_{j_1,i}\left(\sum_{j=1}^{m} w_{j,j_2}x_{j,i} - x_{j_2,i}\right) < \lambda$ (for $w_{j_1,j_2} < 0$), $\left|\sum_{i=1}^{n} x_{j_1,i}\left(\sum_{j=1}^{m} w_{j,j_2}x_{j,i} - x_{j_2,i}\right)\right| < \lambda$ (for $w_{j_1,j_2} = 0$).

Since the optimal value for w_{j_1,j_2} does not depend on $w_{j_1,j}$ for $j \neq j_2$ given $\max_{j=1}^{m} |w_{j_1,j}|$, we can efficiently solve for an entire row at a time.

To solve for one row while keeping all others fixed, we have the following optimization problem:

$$\text{Minimize}_{W_r \in \mathbb{R}^m} \quad \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{m} \left(\left(x_{j,i} - \sum_{k \neq r} w_{k,j} x_{k,i} \right) - w_{r,j} x_{r,i} \right)^2 + \lambda \max_{j=1}^{m} |w_{r,j}|$$

Equivalently:

$$\begin{array}{rc} \text{Minimize}_{W_r \in \mathbb{R}^m} & \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{m} \left(\left(x_{j,i} - \sum_{k \neq r} w_{k,j} x_{k,i} \right) - w_{r,j} x_{r,i} \right)^2 + \lambda M_r \\ \text{Subject to} & \forall j \neq r, \quad |w_{r,j}| \leq M_r \end{array}$$

If there was no penalty term, we could simply set the derivatives to zero to get the unpenalized optimal values:

$$\hat{w}_{r,j} \quad = \quad \frac{-\sum_{i=1}^{n} x_{r,i} \left(\sum_{j_2 \neq r} w_{j_2,j} x_{j_2,i} - x_{j,i} \right)}{\sum_{i=1}^{n} x_{r,i}^2}$$

Now examine the optimal assignments to W_r if M_r is given. Let $\mathcal{I}_r^{(1)} = \{j \neq r : \hat{w}_{r,j} \geq M_r\}$, $\mathcal{I}_r^{(2)} = \{j \neq r : \hat{w}_{r,j} \leq -M_r\}$, and $\mathcal{I}_r^{(3)} = \{j \neq r : |\hat{w}_{r,j}| < M_r\}$. Then using those notations, the optimal values of W_r are:

$$w_{r,j} \quad = \quad \begin{cases} M_r & \text{if } j \in \mathcal{I}_r^{(1)} \\ -M_r & \text{if } j \in \mathcal{I}_r^{(2)} \\ \hat{w}_{r,j} & \text{if } j \in \mathcal{I}_r^{(3)} \end{cases}$$

The objective value for a given M_r is:

$$
\begin{aligned}
J\left(M_r\right) &= \frac{1}{2}\sum_{j\in\mathcal{I}_r^{(3)}}\sum_{i=1}^{n}\left(\sum_{k\neq r}w_{k,j}x_{k,i}+\hat{w}_{r,j}x_{r,i}-x_{j,i}\right)^2+\\
&+ \frac{1}{2}\sum_{j\in\mathcal{I}_r^{(1)}}\sum_{i=1}^{n}\left(\sum_{k\neq r}w_{k,j}x_{k,i}+M_r x_{r,i}-x_{j,i}\right)^2+\\
&+ \frac{1}{2}\sum_{j\in\mathcal{I}_r^{(2)}}\sum_{i=1}^{n}\left(\sum_{k\neq r}w_{k,j}x_{k,i}-M_r x_{r,i}-x_{j,i}\right)^2+\lambda M_r\\
J'\left(M_r\right) &= \sum_{j\in\mathcal{I}_r^{(1)}}\sum_{i=1}^{n}x_{r,i}\left(\sum_{k\neq r}w_{k,j}x_{k,i}-x_{j,i}+M_r x_{r,i}\right)+\\
&+ \sum_{j\in\mathcal{I}_r^{(2)}}\sum_{i=1}^{n}x_{r,i}\left(x_{j,i}-\sum_{k\neq r}w_{k,j}x_{k,i}+M_r x_{r,i}\right)+\lambda\\
&= \left(\left|\left\{j\in\mathcal{I}_r^{(1)}\cup\mathcal{I}_r^{(2)}\right\}\right|\sum_{i=1}^{n}x_{r,i}^2\right)M_r-\\
&- \sum_{j\in\mathcal{I}_r^{(1)}\cup\mathcal{I}_r^{(1)}}\left|\sum_{i=1}^{n}x_{r,i}\left(\sum_{k\neq r}w_{k,j}x_{k,i}-x_{j,i}\right)\right|+\lambda
\end{aligned}
$$

Setting the derivative to zero gives the optimal M_r given $\mathcal{I}_r^{(1)}$, $\mathcal{I}_r^{(2)}$, $\mathcal{I}_r^{(3)}$:

$$
\hat{M}_r = \frac{\sum_{j\in\mathcal{I}_r^{(1)}\cup\mathcal{I}_r^{(1)}}\left|\sum_{i=1}^{n}x_{r,i}\left(\sum_{k\neq r}w_{k,j}x_{k,i}-x_{j,i}\right)\right|-\lambda}{\left|\left\{j\in\mathcal{I}_r^{(1)}\cup\mathcal{I}_r^{(2)}\right\}\right|\sum_{i=1}^{n}x_{r,i}^2}
$$

2.5.8 L_2-regularized linear regression

This method is still linear, but it is discriminative rather than generative. We directly predict the expression values of the unobserved probes from those of the observed ones. For a probeset $\mathcal{S}$, we first subtract the mean $\bar{X}$ from $\boldsymbol{X}$ to get $\tilde{\boldsymbol{X}}$ and solve:

$$
W_{\mathcal{S}} = \left(\tilde{\boldsymbol{X}}_{\mathcal{S}}\tilde{\boldsymbol{X}}_{\mathcal{S}}^T+\lambda I\right)^{-1}\left(\tilde{\boldsymbol{X}}_{\mathcal{S}}\right)\left(\tilde{\boldsymbol{X}}_{-\mathcal{S}}\right)^T
$$

$W\in\mathbb{R}^{l\times m-l}$ is the prediction matrix. The imputation is then: $Y_{-\mathcal{S}}=W_{\mathcal{S}}^T Y_{\mathcal{S}}$.
Given a probeset, the LOOCV likelihood as a function of λ is:

$$
-l\left(\lambda:\mathcal{D}\right) = \sum_{i=1}^{n}\sum_{j\notin\mathcal{S}}\left(X_{j,i}-\tilde{\boldsymbol{X}}_{j,-i}\tilde{\boldsymbol{X}}_{\mathcal{S},-i}^T\left(\tilde{\boldsymbol{X}}_{\mathcal{S},-i}\tilde{\boldsymbol{X}}_{\mathcal{S},-i}+\lambda I\right)^{-1}X_{\mathcal{S},i}\right)^2
$$

Let $\boldsymbol{X}_{\mathcal{S}}\boldsymbol{X}_{\mathcal{S}}^T = U[\mathcal{S}]S[\mathcal{S}]U[\mathcal{S}]^T$ be the eigen-decomposition of $\boldsymbol{X}_{\mathcal{S}}$. Then $\tilde{\boldsymbol{X}}_{\mathcal{S},-i}\tilde{\boldsymbol{X}}_{\mathcal{S},-i}^T = \boldsymbol{X}_{\mathcal{S}}\boldsymbol{X}_{\mathcal{S}}^T - X_{\mathcal{S},i}X_{\mathcal{S},i}^T - \frac{1}{n-1}\left(\sum_{i'\neq i} X_{\mathcal{S},i'}\right)\left(\sum_{i'\neq i} X_{\mathcal{S},i'}\right)^T$.

Using the Sherman-Morrison formula:

$$
\begin{aligned}
M_i[\lambda] &= \left(\tilde{\boldsymbol{X}}_{\mathcal{S},-i}\tilde{\boldsymbol{X}}_{\mathcal{S},-i} + \lambda I\right)^{-1} \\
&= \left(\boldsymbol{X}_{\mathcal{S}}\boldsymbol{X}_{\mathcal{S}}^T + \lambda I - X_{\mathcal{S},i}X_{\mathcal{S},i}^T - \frac{1}{n-1}\left(\sum_{i'\neq i} X_{\mathcal{S},i'}\right)\left(\sum_{i'\neq i} X_{\mathcal{S},i'}\right)^T\right)^{-1} \\
R_i[\lambda]^{-1} &= \left(\boldsymbol{X}_{\mathcal{S}}\boldsymbol{X}_{\mathcal{S}}^T + \lambda I - X_{\mathcal{S},i}X_{\mathcal{S},i}^T\right)^{-1} \\
&= \left(\boldsymbol{X}_{\mathcal{S}}\boldsymbol{X}_{\mathcal{S}}^T + \lambda I\right)^{-1}\left(I + \frac{X_{\mathcal{S},i}X_{\mathcal{S},i}^T}{1 - X_{\mathcal{S},i}^T\left(\boldsymbol{X}_{\mathcal{S}}\boldsymbol{X}_{\mathcal{S}}^T + \lambda I\right)^{-1}X_{\mathcal{S},i}}\left(\boldsymbol{X}_{\mathcal{S}}\boldsymbol{X}_{\mathcal{S}}^T + \lambda I\right)^{-1}\right) \\
M_i[\lambda] &= \left(R_i[\lambda] + \left(\sum_{i'\neq i} X_{\mathcal{S},i'}\right)\left(\frac{1}{1-n}\sum_{i'\neq i} X_{\mathcal{S},i'}\right)^T\right)^{-1} \\
&= R_i[\lambda]^{-1}\left(I + \frac{\left(\sum_{i'\neq i} X_{\mathcal{S},i'}\right)\left(\sum_{i'\neq i} X_{\mathcal{S},i'}\right)^T}{n - 1 - \left(\sum_{i'\neq i} X_{\mathcal{S},i'}\right)^T R_i[\lambda]^{-1}\left(\sum_{i'\neq i} X_{\mathcal{S},i'}\right)}R_i[\lambda]^{-1}\right) \\
l(\lambda : \mathcal{D}) &= \sum_{i=1}^{n}\sum_{j\notin\mathcal{S}}\left(X_{j,i} - \tilde{\boldsymbol{X}}_{j,-i}\tilde{\boldsymbol{X}}_{\mathcal{S},-i}^T M_i[\lambda]\, X_{\mathcal{S},i}\right)^2
\end{aligned}
$$

This allows efficient computation of the LOOCV likelihood for r different values of λ at the cost of $O\left(rn\left|\mathcal{S}\right|^2\right)$, given the SVD of $\boldsymbol{X}_{\mathcal{S}}$.

For efficient computation, we will need to compute the SVD of $\boldsymbol{X}_{\mathcal{S}}$ incrementally. Let $X = USV^T$. We want the SVD of $\begin{pmatrix} X \\ y \end{pmatrix}$. Let:

$$
\begin{aligned}
U_1 &= \begin{pmatrix} U & 0 \\ 0 & 1 \end{pmatrix} \\
k &= \left\|y\left(I - VV^T\right)\right\|_2 \\
J &= \frac{y\left(I - VV^T\right)}{k} \\
Q_1 &= \begin{pmatrix} S & 0 \\ yV & k \end{pmatrix} \\
V_1^T &= \begin{pmatrix} V^T \\ J \end{pmatrix}
\end{aligned}
$$

Then:

$$
\begin{aligned}
U_1 Q_1 V_1^T &= \begin{pmatrix} US & 0 \\ yV & k \end{pmatrix} \begin{pmatrix} V^T \\ J \end{pmatrix} \\
&= \begin{pmatrix} USV^T \\ yVV^T + y - yVV^T \end{pmatrix} = \begin{pmatrix} X \\ y \end{pmatrix}
\end{aligned}
$$

Since Q_1 is not diagonal, we need to diagonalize it. The matrix $Q_1 Q_1^T = \begin{pmatrix} S & 0 \\ yV & k \end{pmatrix} \begin{pmatrix} S & V^T y^T \\ 0 & k \end{pmatrix} = \begin{pmatrix} S^2 & SV^T y^T \\ yVS & yVV^T y^T + k^2 \end{pmatrix}$ is an arrowhead matrix and we use this property as previously suggested [2] to compute its eigen-decomposition $Q_1 Q_1^T = U_2 S_2^2 U_2^T$. Then let $V_2 = Q_1^T U_2 S_2^{-1}$, so that $U_2 S_2 V_2^T = U_2 S_2 S_2^{-1} U_2^T Q_1 = Q_1$.

Now let $U_3 = U_1 U_2$, $S_3 = S_2$ and $V_3 = V_1 V_2$ to get:

$$
U_3 S_3 V_3^T = U_1 U_2 S_2 V_2^T V_1^T = U_1 Q_1 V_1^T = \begin{pmatrix} X \\ y \end{pmatrix}
$$

This allows us to update the SVD of $\boldsymbol{X}_{\mathcal{S}}$ with a new probe to the SVD of $\boldsymbol{X}_{\mathcal{S}\cup\{j\}}$ in $O\left(|\mathcal{S}|^2\right)$.

To select a probeset, we apply a greedy procedure, choosing at each iteration the probe that maximizes the likelihood with λ fixed. Assume we already computed $Q_{-i} = \left(\tilde{\boldsymbol{X}}_{\mathcal{S},-i} \tilde{\boldsymbol{X}}_{\mathcal{S},-i}^T + \lambda I\right)^{-1}$. Then:

$$
\begin{aligned}
A[j] &= \left(\tilde{\boldsymbol{X}}_{\mathcal{S}\cup\{j\},-i} \tilde{\boldsymbol{X}}_{\mathcal{S}\cup\{j\},-i}^T + \lambda I\right) = \begin{pmatrix} \left(\tilde{\boldsymbol{X}}_{\mathcal{S},-i} \tilde{\boldsymbol{X}}_{\mathcal{S},-i}^T + \lambda I\right) & \tilde{\boldsymbol{X}}_{\mathcal{S},-i} \tilde{X}_{j,-i}^T \\ \tilde{X}_{j,-i} \tilde{\boldsymbol{X}}_{\mathcal{S},-i}^T & \tilde{X}_{j,-i} \tilde{X}_{j,-i}^T + \lambda \end{pmatrix} \\
v &= \tilde{\boldsymbol{X}}_{\mathcal{S},-i} \tilde{X}_{j,-i}^T \\
u &= Qv \\
d &= \frac{1}{\tilde{X}_{j,-i} \tilde{X}_{j,-i}^T + \lambda - v^T u} \\
A[j]^{-1} &= \begin{pmatrix} Q + duu^T & -du \\ -du^T & d \end{pmatrix}
\end{aligned}
$$

Now solving for the regularized regression weights:

$$
\begin{aligned}
A[j]^{-1} \begin{pmatrix} \tilde{\boldsymbol{X}}_{\mathcal{S},-i} \\ \tilde{X}_{j,-i} \end{pmatrix} &= \begin{pmatrix} Q\tilde{\boldsymbol{X}}_{\mathcal{S},-i} + du\left(u^T \tilde{\boldsymbol{X}}_{\mathcal{S},-i} - \tilde{X}_{j,-i}\right) \\ -d\left(u^T \tilde{\boldsymbol{X}}_{\mathcal{S},-i} - \tilde{X}_{j,-i}\right) \end{pmatrix} \\
W[j] &= A[j]^{-1} \begin{pmatrix} \tilde{\boldsymbol{X}}_{\mathcal{S},-i} \\ \tilde{X}_{j,-i} \end{pmatrix} \tilde{\boldsymbol{X}}_{-\mathcal{S}-\{j\},-i}^T \\
&= \begin{pmatrix} Q\tilde{\boldsymbol{X}}_{\mathcal{S},-i} + du\left(u^T \tilde{\boldsymbol{X}}_{\mathcal{S},-i} - \tilde{X}_{j,-i}\right) \\ -d\left(u^T \tilde{\boldsymbol{X}}_{\mathcal{S},-i} - \tilde{X}_{j,-i}\right) \end{pmatrix} \tilde{\boldsymbol{X}}_{-\mathcal{S}-\{j\},-i}^T = \begin{pmatrix} U[j] \\ u[j] \end{pmatrix}
\end{aligned}
$$

The imputation of probe j_1 using $\mathcal{S} \cup \{j_2\}$ would then be:

$$
\begin{aligned}
r\left[j\right] &= d\left(u^T \tilde{\boldsymbol{X}}_{\mathcal{S},-i} - \tilde{X}_{j,-i}\right) \\
y_{j_1,j_2} &= U[j_2]_{j_1}^T \tilde{X}_{\mathcal{S},i} + u[j_2]_{j_1} \tilde{X}_{j_2,i} \\
&= \tilde{X}_{j_1,-i}\left(\left(r[j_2]^T u^T + \tilde{\boldsymbol{X}}_{\mathcal{S},-i}^T Q\right)\tilde{X}_{\mathcal{S},i} - r[j_2]^T \tilde{X}_{j_2,i}\right) \\
&= \tilde{X}_{j_1,-i} p\left[j_2\right] \\
p\left[j_2\right] &= \left(r[j_2]^T u^T + \tilde{\boldsymbol{X}}_{\mathcal{S},-i}^T Q\right)\tilde{X}_{\mathcal{S},i} - r[j_2]^T \tilde{X}_{j_2,i}
\end{aligned}
$$

The corresponding error:

$$
\begin{aligned}
t\left[j_2\right] &= r\left[j_2\right] \tilde{\boldsymbol{X}}_{-\mathcal{S},-i}^T \tilde{X}_{-\mathcal{S},i} \\
\hat{\epsilon}\left[j_2\right] &= \sum_{j_1 \notin \mathcal{S}} \left(y_{j_1,j_2} - \tilde{X}_{j_1,i}\right)^2 \\
&= \tilde{X}_{-\mathcal{S},i}^T \tilde{X}_{-\mathcal{S},i} - 2 y_{j_2}^T \tilde{X}_{-\mathcal{S},i} + y_{j_2}^T y_{j_2} \\
&= \tilde{X}_{-\mathcal{S},i}^T \tilde{X}_{-\mathcal{S},i} + 2\tilde{X}_{j_2,i} t\left[j_2\right] - 2\tilde{X}_{\mathcal{S},i}^T Q_{-i} \tilde{\boldsymbol{X}}_{\mathcal{S},-i}\left(\tilde{X}_{j2,-i}^T t\left[j_2\right] + 1\right) + y_{j_2}^T y_{j_2} \\
y_{j_2}^T y_{j_2} &= \sum_{j_1 \notin \mathcal{S}} y_{j_1,j_2}^2 = \sum_{j_1 \notin S} \tilde{X}_{j_1,-i} p\left[j_2\right] p\left[j_2\right]^T \tilde{X}_{j_1,-i}^T \\
&= p\left[j_2\right]^T \tilde{\boldsymbol{X}}_{-\mathcal{S},-i}^T \tilde{\boldsymbol{X}}_{-\mathcal{S},-i} p\left[j_2\right]
\end{aligned}
$$

2.5.9 Group lasso regression

This method uses regularization for selection as well as parameter estimation. Let $W \in \mathbb{R}^{m \times m}$ be the imputation matrix, such that $\mathcal{S}\left[W\right] = \{1 \leq j \leq m : \sum_{k=1}^{m} W_{j,k}^2 > 0\}$.

Then given W, the likelihood term is: $\sum_{i=1}^{n} \left\|W^T X_i - X_i\right\|_2^2$. We add a group lasso penalty term $\lambda \sum_{j=1}^{m} \sqrt{\sum_{k=1}^{m} W_{j,k}^2}$. Then the optimization goal is:

$$
\text{Minimize} \quad \frac{1}{2}\sum_{i=1}^{n} \left\|W^T X_i - X_i\right\|_2^2 + \lambda \sum_{j=1}^{m} \sqrt{\sum_{k=1}^{m} W_{j,k}^2} \tag{6}
$$

This goal is convex in W and we solve it by coordinate descent on rows of W. When all the rows are fixed except the j'th row, the goal is:

$$
J\left(W_j\right) = \text{const} + \frac{1}{2}\sum_{i=1}^{n} \left\|W_j^T x_{j,i} - \left(X_i - W_{-j}^T X_{-j,i}\right)\right\|_2^2 + \sqrt{\sum_{k=1}^{m} W_{j,k}^2}
$$

Let $R_i = X_i - W_{-j}^T X_{-j,i}$ be the matrix of residuals. Take the gradient of J:

$$
\begin{aligned}
\frac{\partial J(W_j)}{\partial W_j} &= \sum_{i=1}^{n} x_{j,i}\left(x_{j,i}W_j - R_i\right) + \frac{\lambda W_j}{\sqrt{\sum_{k=1}^{m} W_{j,k}^2}} \\
\sum_{i=1}^{n} x_{j,i}R_i &= \left(\sum_{i=1}^{n} x_{j,i}^2\right) W_j + \frac{\lambda}{\|W_j\|_2} W_j \\
W_j &= \frac{\sum_{i=1}^{n} x_{j,i}R_i}{\left(\sum_{i=1}^{n} x_{j,i}^2\right) + \frac{\lambda}{\|W_j\|_2}} \\
W_j &= \|W_j\|_2 \frac{\sum_{i=1}^{n} x_{j,i}R_i}{\|W_j\|_2 \left(\sum_{i=1}^{n} x_{j,i}^2\right) + \lambda} \\
\|W_j\|_2 &= \frac{\|W_j\|_2}{\|W_j\|_2 \sum_{i=1}^{n} x_{j,i}^2 + \lambda} \left\| \sum_{i=1}^{n} x_{j,i}R_i \right\|_2 \\
\|W_j\|_2 &= \frac{\left\| \sum_{i=1}^{n} x_{j,i}R_i \right\|_2 - \lambda}{\sum_{i=1}^{n} x_{j,i}^2}
\end{aligned}
$$

If $\left\| \sum_{i=1}^{n} x_{j,i}R_i \right\|_2 \leq \lambda$, then $W_j = 0$. Otherwise,

$$
\begin{aligned}
W_j &= \frac{\|W_j\|_2 \sum_{i=1}^{n} x_{j,i}R_i}{\left\| \sum_{i=1}^{n} x_{j,i}R_i \right\|} \\
&= \frac{\|W_j\|_2}{\|W_j\|_2 \sum_{i=1}^{n} x_{j,i}^2 + \lambda} \left\| \sum_{i=1}^{n} x_{j,i}R_i \right\|_2 \frac{\sum_{i=1}^{n} x_{j,i}R_i}{\left\| \sum_{i=1}^{n} x_{j,i}R_i \right\|} \\
&= \frac{\|W_j\|_2}{\|W_j\|_2 \sum_{i=1}^{n} x_{j,i}^2 + \lambda} \sum_{i=1}^{n} x_{j,i}R_i \\
&= \frac{\|W_j\|_2}{\left\| \sum_{i=1}^{n} x_{j,i}R_i \right\|_2} \sum_{i=1}^{n} x_{j,i}R_i \\
&= \frac{\sum_{i=1}^{n} x_{j,i}R_i}{\sum_{i=1}^{n} x_{j,i}^2} \left(1 - \frac{\lambda}{\left\| \sum_{i=1}^{n} x_{j,i}R_i \right\|_2}\right)
\end{aligned}
$$

We can combine both cases and write:

$$W_j = \frac{\sum_{i=1}^{n} x_{j,i} R_i}{\sum_{i=1}^{n} x_{j,i}^2} \left(1 - \frac{\lambda}{\left\| \sum_{i=1}^{n} x_{j,i} R_i \right\|_2} \right)_+ \tag{7}$$

For each number of genes we assign a value of λ. For $l = 0$, we need $\left\| \sum_{i=1}^{n} x_{j,i} X_i \right\|_2 \leq \lambda$ for all $1 \leq j \leq m$. Hence, $\lambda[0] = \max_{1 \leq j \leq m} \left\| \sum_{i=1}^{n} x_{j,i} X_i \right\|_2$. Now assume we have $\mathcal{S}[l]$, $\lambda[l]$ and a solution $W[l]$ for that value of λ. We compute the residuals R_i with that value of W, and choose $\lambda[l+1] = \max_{j \notin S} \left\| \sum_{i=1}^{n} x_{j,i} R_i \right\|_2$. This gives the highest value of λ such that below this value, keeping all other weights fixed, a new probe is introduced. Since the optimal value for this probe is still 0 when λ is computed as described above, we set $\lambda[l+1] = \alpha \max_{j \notin S} \left\| \sum_{i=1}^{n} x_{j,i} R_i \right\|_2$ for some $\alpha < 1$ ($\alpha = 0.99$ in our experiments).

2.5.10 GLL2: Group Lasso selection with L_2 regression

In Section 2.5.9 we described how to use group lasso regularization for both selection and shrinkage. Since the regularization parameter is chosen to yield a prespecified number of probes, it is not optimized for best generalization to unobserved data, and in fact it is quite likely that its chosen values are very unsuitable for good generalization of predictions.

For this reason, in this method we use the group lasso regularization described in Section 2.5.9 for selection, but once the probes have been selected, we use L_2-regularized linear regression to impute the missing values. The regularization parameter for the L_2-regularized regression is chosen using leave-one-out cross-validation.

2.6 Modular methods

2.6.1 Covariance estimation with module structure

In this method, we combine any of the other methods for covariance estimation with exploitation of the modular structure of gene expression.

Given a partition of the probes into modules $M_1, \ldots, M_{n_M}$, let C_j be the module to which probe j is assigned, then we model the vector of gene expression X as $X_j = Z_{C_j} + \nu_j + \epsilon_j$, where $Z_1, \ldots, Z_{n_M}$ are the module means and $Z \sim \mathcal{N}(\xi; \Psi)$, ν_j is the difference between the expression of probe j and the mean of its module and $C_{j_1} \neq C_{j_2} \Rightarrow \nu_{j_1} \perp \nu_{j_2}$, and $\epsilon \sim \mathcal{N}\left(0, \sigma^2 I\right)$. In words, the mean module expression is modeled as a multivariate Gaussian, and the differential expression of the probes within each module from the module mean is another multivariate Gaussian with zero mean. This differential expression is independent of the differential expression of all probes from different modules. Hence, the dependencies between the expression values of probes from different modules are only through the dependencies between the means of their modules. Additionally there is independent Gaussian noise added to each probe.

The distribution over the probes is then $\mathcal{N}(\mu; \Sigma)$ where μ is the empirical mean and Σ is the full estimated covariance matrix given σ^2 (variance of the noise), ξ (module mean expression values), Ψ (estimate covariance of module means), and $\Phi_1, \ldots, \Phi_{n_M}$ (estimate covariance matrices for $\nu_{M_i} - \xi_i$, the intra-module deviation from the module mean): the sub-matrix corresponding to the covariance within a module M_j is $\Psi_{j,j} \mathbf{1}_{|M_j|} \mathbf{1}_{|M_j|}^T + \Phi_j + \sigma^2 I$, and the sub-matrix corresponding to the covariance between modules M_{j_1} and M_{j_2} is $\Psi_{j_1,j_2} \mathbf{1}_{|M_{j_1}|} \mathbf{1}_{|M_{j_2}|}^T$.

This general model dictates the form of the overall Gaussian distribution over all genes, given the module assignments and the distributions of Z and ν. We also need to specify how to choose the modules $M_1, \ldots, M_{n_M}$ and how to estimate the covariances over Z and ν.

One way of evaluating a particular module assignment is to ignore ν and use the Bayesian score $P(X|C_1, \ldots, C_m)$ which integrates over the hidden variables Z and ϵ. Due to the structure of the model, if we ignore ν this results in a closed-form expression which is the product of the individual marginal module probabilities $P(X_{M_r})$. Since for $j \in M_r$ we have $X_j - Z_r \sim \mathcal{N}\left(0, \sigma^2\right)$, then $\mathbb{E}[X_j] = \mathbb{E}_{Z_r}[\mathbb{E}[X_j|Z_r]] = \mathbb{E}_{Z_r}[Z_r] = \xi_r$, $\mathrm{Var}[X_j] = \mathrm{Var}[Z_r] + \sigma^2 = \Psi_{r,r} + \sigma^2$ and for $j_1, j_2 \in M_r$ we have $\mathrm{Cov}[X_{j_1}; X_{j_2}] = \Psi_{r,r}$. Hence, the marginal distribution over X_{M_r} is $\mathcal{N}(\xi_r; \Sigma_r)$ where $\Sigma_r = \Psi_{r,r}\mathbf{1}\mathbf{1}^T + \sigma^2 I$ and its inverse is $J_r = \frac{1}{\sigma^2}\left(I - \frac{\Psi_{r,r}}{\sigma^2 + |M_r|\Psi_{r,r}}\mathbf{1}\mathbf{1}^T\right)$ which allows us to compute $P(X) = (2\pi)^{m/2}\prod_{r=1}^{n_M}\sqrt{|J_r|}\exp\left(-\frac{1}{2}\left(X_{M_r} - \xi_r\right)^T J_r\left(X_{M_r} - \xi_r\right)\right)$. Ignoring ν means we only need to estimate expectations and variances, but no covariances.

Since σ^2 is not known, we approximate marginalizing over it by doing Metropolis Hastings with an initialization of $\left(\sigma^2\right)^{(0)} = 1$ and two proposal moves: $\left(\sigma^2\right)^{(t+1)} = \gamma\left(\sigma^2\right)^{(t)}$ and $\left(\sigma^2\right)^{(t+1)} = \gamma^{-1}\left(\sigma^2\right)^{(t)}$, which have equal probabilities. Then we use the Bayesian score described above to compute an acceptance probability.

For a given module assignment, we can use any of the covariance estimation methods to estimate Ψ, the covariance matrix for the module means, and $\Sigma_1, \ldots, \Sigma_{n_M}$, the covariance matrices for the differential expression from the module mean of probes within a module. The simplest implementation uses K-means clustering to generate a single module assignment. We can use the Bayesian score described above to compare within different clustering results, for different runs of K-means with the same or over a range of values for K. We can also use a search method to optimize the Bayesian score directly. For improved robustness, we use 100 random restarts of K-means and take the one with the best K-means objective score. We compute the Bayesian score for the best module assignment for each K ($30 \leq K \leq 100$ in jumps of 5) and use the assignment with the best score. The value of σ^2 is chosen to maximize the Bayesian score. We refer to this as "K-Means Module Assignment". In practice, for our experiments we used 100 random K-means restarts for each number of modules that we tested.

A more refined approach is to sample module assignments using the Bayesian score, and for each module assignments, estimate the corresponding full covariance matrix (by estimating Ψ and $\Sigma_1, \ldots, \Sigma_{n_M}$), and finally average over these covariance matrices. This may result in a better estimate of the full covariance matrix which takes into account the uncertainty of the module assignments.

The sampling is done using Metropolis-Hastings, and the Markov chain is a mixture of two kernels, where in each step the kernel to be used is chosen randomly with a prespecified probability. The first Markov chain kernel generates proposal moves that move one probe between modules. A probe is chosen uniformly at random, and then either joined to one of the other modules (chosen at random) or (if it is not the only probe in its module) is moved to a new module with the same probability as being moved to any of the existing modules. For this Markov chain, the proposal probabilities satisfy $\mathcal{Q}(\Pi \to \Pi') = \mathcal{Q}(\Pi' \to \Pi)$.

The second Markov chain kernel is based on moves that split and join modules. The type of move is chosen at random, where the probability of a join move is p (or 0 if there is only one module, or 1 if every probe is in its own module). A join move consists of choosing two modules uniformly at random and joining them. The corresponding proposal probability (for joining two modules in Π' to a single module in Π) is $\mathcal{Q}(\Pi' \to \Pi) = \frac{2p}{n_M(n_M - 1)}$. A split move consists of choosing a module with at least two probes uniformly at random, and reassigning all its probes at random to one of two new modules, as long as at least one probe is assigned to each of the new modules. The corresponding proposal probability (for splitting module i in Π to two non-empty modules in Π') is $\mathcal{Q}(\Pi \to \Pi') = \frac{1-p}{|\{i' : |M_{i'}| \geq 2\}|}\frac{2}{2^{|M_i|} - 2} = \frac{1}{|\{i' : |M_{i'}| \geq 2\}|\left(2^{|M_i|-1} - 1\right)}$.

n_s samples are generated using the Metropolis-Hastings algorithm, and the n_b first samples are thrown away. For each of the remaining samples, a full covariance matrix is estimated and the final covariance matrix is the average of these matrices.

In practice, we did not use this sampling method due to its extremely long running times.

2.6.2 Efficient probe selection with modules

Let $v_i \in \mathbb{R}^m$ be a column vector such that $(v_i)_j = \mathbf{1}[C_j = i]$. In other words, it is the indicator function of the j'th module. Let $V \in \mathbb{R}^{m \times n_M}$ be the matrix with $v_1, \ldots, v_{n_M}$ as its columns. The full covariance matrix is: $\Sigma = \mathrm{Diag}(\Phi_1, \ldots, \Phi_{n_M}) + \sigma^2 I + \sum_{i_1=1}^{n_M} \sum_{i_2=1}^{n_M} v_{i_1} v_{i_2}^T \Psi_{i_1,i_2} = \mathrm{Diag}\left(\Phi_1 + \sigma^2 I, \ldots, \Phi_{n_M} + \sigma^2 I\right) + V \Psi V^T$. Let $A = \mathrm{Diag}\left(\Phi_1 + \sigma^2 I, \ldots, \Phi_{n_M} + \sigma^2 I\right)$ and $A^{-1} = \mathrm{Diag}\left(\left(\Phi_1 + \sigma^2 I\right)^{-1}, \ldots, \left(\Phi_{n_M} + \sigma^2 I\right)^{-1}\right)$. Then by the Woodbury formula:

$$\Sigma^{-1} = A^{-1} - A^{-1} V \left(\Psi^{-1} + V^T A^{-1} V\right)^{-1} V^T A^{-1}$$

For two given probes j_1 and j_2, their conditional covariance given $\mathcal{S}$ is $\Sigma_{j_1,j_2} - \Sigma_{j_1,\mathcal{S}} \Sigma_{\mathcal{S},\mathcal{S}}^{-1} \Sigma_{\mathcal{S},j_2}$. Let $V_\mathcal{S}$ be the module assignment indicator matrix as V above, restricted to $\mathcal{S}$, then $\Sigma_{\mathcal{S},j} = V_\mathcal{S} \Psi_{C_j}^T + u_j$ where $\Psi_{C_j}^T$ is the row of Ψ that corresponds to the module to which probe j belongs, and u_j is a vector that corresponds to the within-module covariance of j with elements of $\mathcal{S}$ (so $(u_j)_k = 0$ for elements k that are not in the same module as j). The conditional covariance is, then:

$$\Sigma_{j_1,j_2} - \left(\Psi_{C_{j_1}} V_\mathcal{S}^T + u_{j_1}^T\right) \left(A_\mathcal{S}^{-1} - A_\mathcal{S}^{-1} V_\mathcal{S} \left(\Psi^{-1} + V_\mathcal{S}^T A_\mathcal{S}^{-1} V_\mathcal{S}\right)^{-1} V_\mathcal{S}^T A_\mathcal{S}^{-1}\right) \left(V_\mathcal{S} \Psi_{C_{j_2}}^T + u_{j_2}\right)$$

The imputation weights for probe j belonging to module c are:

$$\left(\Psi_c V_\mathcal{S}^T + u_j^T\right) \left(A_\mathcal{S}^{-1} - A_\mathcal{S}^{-1} V_\mathcal{S} \left(\Psi^{-1} + V_\mathcal{S}^T A_\mathcal{S}^{-1} V_\mathcal{S}\right)^{-1} V_\mathcal{S}^T A_\mathcal{S}^{-1}\right)$$

Since $A_\mathcal{S}^{-1}$ is block-diagonal, $A_\mathcal{S}^{-1} V_\mathcal{S}$ is a $|\mathcal{S}| \times n_M$ matrix where the i'th column is the sum of the columns of $A_\mathcal{S}$ corresponding to the i'th module, and therefore is nonzero only in rows corresponding to the i'th module. $M_\mathcal{S} = V_\mathcal{S}^T A_\mathcal{S}^{-1} V_\mathcal{S}$ is a diagonal $n_M \times n_M$ matrix with $M_{i,i} = \sum_{j_1 \in \mathcal{S}: C_{j_1} = i} \sum_{j_2 \in \mathcal{S}: C_{j_2} = i} \left(A_\mathcal{S}^{-1}\right)_{j_1,j_2}$.

$u_j^T A_\mathcal{S}^{-1} V_\mathcal{S}$ is a row vector of n_M elements, all of them zeros except the C_j'th element, which is equal to

$$\sum_{j_1 \in \mathcal{S}: C_{j_1} = C_j} \sum_{j_2 \in \mathcal{S}: C_{j_2} = C_j} \left(\Phi_{C_j}\right)_{j,j_1} \left(A_\mathcal{S}^{-1}\right)_{j_1,j_2}$$

Hence:

$$u_{j_1}^T A_\mathcal{S}^{-1} V_\mathcal{S} \Psi_{C_{j_2}}^T = \Psi_{C_{j_1},C_{j_2}} \sum_{k_1 \in \mathcal{S}: C_{k_1} = C_{j_1}} \sum_{k_2 \in \mathcal{S}: C_{k_2} = C_{j_1}} \left(\Phi_{C_j}\right)_{j_1,k_1} \left(A_\mathcal{S}^{-1}\right)_{k_1,k_2}$$

So we compute for each module the sum of columns of $A_\mathcal{S}^{-1}$ corresponding to its probes, and then for each probe in the module we compute the weighted sum of the elements of this sum where the weights are taken from the internal module covariance matrix. Let r_j denote this scalar. This is done in a total of $\sum_{i=1}^{n_M} |M_i \cap \mathcal{S}|^2$ operations. Then $u_{j_1}^T A_\mathcal{S}^{-1} V_\mathcal{S} \Psi_{C_{j_2}}^T$ is simply r_{j_1}, multiplied by the inter-module covariance of C_{j_1} and C_{j_2}.

$u_{j_1}^T A_{\mathcal{S}}^{-1} u_{j_2}$ is nonzero only when $C_{j_1} = C_{j_2} = c$ and is equal to $\sum_{k_1 \in \mathcal{S}: C_{k_1} = c} \sum_{k_2 \in \mathcal{S}: C_{k_2} = c} (\Phi_c)_{j_1,k_1} \left(A_{\mathcal{S}}^{-1}\right)_{k_1,k_2} (\Phi_c)_{j_2,k_2}$. For each module, this is done in $|M_i \cap \mathcal{S}|^3$ operations.

Similarly, $u_{j_1}^T A_{\mathcal{S}}^{-1} V_{\mathcal{S}} \left(\Psi^{-1} + V_{\mathcal{S}}^T A_{\mathcal{S}}^{-1} V_{\mathcal{S}}\right)^{-1} V_{\mathcal{S}}^T A_{\mathcal{S}}^{-1} u_{j_2}$ is equal to $r_{j_1} r_{j_2} \left(\Psi^{-1} + V_{\mathcal{S}}^T A_{\mathcal{S}}^{-1} V_{\mathcal{S}}\right)^{-1}_{C_{j_1}, C_{j_2}}$. Also, $\Psi_{C_{j_1}} V_{\mathcal{S}}^T A_{\mathcal{S}}^{-1} V_{\mathcal{S}} \left(\Psi^{-1} + V_{\mathcal{S}}^T A_{\mathcal{S}}^{-1} V_{\mathcal{S}}\right)^{-1} V_{\mathcal{S}}^T A_{\mathcal{S}}^{-1} V_{\mathcal{S}} \Psi_{C_{j_2}}^T$ is equal to

$$\Psi_{C_{j_1}} M \left(\Psi^{-1} + V_{\mathcal{S}}^T A_{\mathcal{S}}^{-1} V_{\mathcal{S}}\right)^{-1} V_{\mathcal{S}}^T \left(A_{\mathcal{S}}\right)^{-1}_{C_{j_1}, C_{j_2}} M \Psi_{C_{j_2}}^T$$

and so can be computed in $O\left(n_M^3\right)$ for all pairs of modules.

To efficiently choose the t'th probe, then:

1. Compute $B_c^{(t)} = \sum_{j \in \mathcal{S}: C_j = c} \left(A_{\mathcal{S}^{(t)}}^{-1}\right)_j$ for $c = 1, \ldots, n_M$. Complexity: $O\left(\sum_{i=1}^{n_M} \left|\mathcal{S}^{(t)} \cap M_i\right|^2\right)$.
2. Compute $G^{(t)} = V_{\mathcal{S}^{(t)}}^T A_{\mathcal{S}^{(t)}}^{-1} V_{\mathcal{S}^{(t)}}$ using $G_{i,i}^{(t)} = \sum_{j_1 \in \mathcal{S}^{(t)}: C_{j_1} = i} B_{C_{j_2}}^{(t)}$. Complexity: $O(t)$.
3. Compute $H^{(t)} = \left(\Psi^{-1} + G^{(t)}\right)^{-1}$. Complexity: $O\left(n_M^3\right)$. This update can be done incrementally, which brings the complexity down to $O\left(n_M^2\right)$.
4. Compute $L^{(t)} = \Psi\left(G^{(t)} - G^{(t)} H^{(t)} G^{(t)}\right)\Psi$. Complexity: $O\left(n_M^3\right)$.
5. Compute $r_j^{(t)} = \sum_{k \in \mathcal{S}^{(t)} \cap M_{C_j}} \left(\Phi_{C_j}\right)_{j,k} B_{k,C_j}^{(t)}$ for all j. Complexity: $O\left(\sum_{i=1}^{n_M} |M_i| \left|\mathcal{S}^{(t)} \cap M_i\right|\right)$.
6. Compute $J^{(t)} = H^{(t)} G^{(t)} \Psi$. Complexity: $O\left(n_M^3\right)$.
7. Compute $D_i^{(t)} = (\Phi_i)_{M_i \setminus \mathcal{S}^{(t)}, M_i \cap \mathcal{S}^{(t)}} A^{-1}_{\mathcal{S}^{(t)} \cap M_i}$. Complexity: $O\left(\sum_{i=1}^{n_M} |M_i| \left|\mathcal{S}^{(t)} \cap M_i\right|^2\right)$.
8. Compute $K_i^{(t)} = D_i^{(t)} (\Phi_i)_{M_i \cap \mathcal{S}^{(t)}, M_i \setminus \mathcal{S}^{(t)}}$. Complexity: $O\left(\sum_{i=1}^{n_M} |M_i|^2 \left|\mathcal{S}^{(t)} \cap M_i\right|\right)$.
9. Compute $a_c^{(t)} = \sum_{j \in M_c \setminus \mathcal{S}^{(t)}} r_j^{(t)}$ and $b_c^{(t)} = \sum_{j \in M_c \setminus \mathcal{S}^{(t)}} \left(r_j^{(t)}\right)^2$ for $c = 1, \ldots, n_M$. Complexity: $O(m)$.
10. For j_1 and j_2 such that $C_{j_1} \neq C_{j_2}$, we have $\sigma^{(t)}[j_1; j_2] = \Psi_{C_{j_1}, C_{j_2}} - L^{(t)}_{C_{j_1}, C_{j_2}} + r_{j_1}^{(t)} \left(J^{(t)}_{C_{j_1}, C_{j_2}} - \Psi_{C_{j_1}, C_{j_2}}\right) + r_{j_2}^{(t)} \left(J^{(t)}_{C_{j_2}, C_{j_1}} - \Psi_{C_{j_1}, C_{j_2}}\right) + r_{j_1}^{(t)} H^{(t)}_{C_{j_1}, C_{j_2}} r_{j_2}^{(t)}$. Let $\alpha_{j_1,c} = \Psi_{C_{j_1},c} - L^{(t)}_{C_{j_1},c} + r_{j_1}^{(t)} \left(J^{(t)}_{C_{j_1},c} - \Psi_{C_{j_1},c}\right)$, $\beta_{j_1,c} = J^{(t)}_{c,C_{j_1}} - \Psi_{C_{j_1},c} + r_{j_1}^{(t)} H^{(t)}_{C_{j_1},c}$. Then $\sigma^{(t)}[j_1; j_2] = \alpha_{j_1,C_{j_2}} + \beta_{j_1,C_{j_2}} r_{j_2}^{(t)}$ and for $c \neq C_{j_1}$, $\sum_{j_2 \in M_c \setminus \mathcal{S}^{(t)}} \sigma^{(t)}[j_1; j_2]^2 = \alpha_{j_1,c}^2 \left|M_c \setminus \mathcal{S}^{(t)}\right| + 2\alpha_{j_1,c}\beta_{j_1,c} \sum_{j_2 \in M_c \setminus \mathcal{S}^{(t)}} r_{j_2}^{(t)} + \beta_{j_1,c}^2 \sum_{j_2 \in M_c \setminus \mathcal{S}^{(t)}} \left(r_{j_2}^{(t)}\right)^2$
11. The conditional covariance between probes j_1 and j_2 given $\mathcal{S}^{(t)}$ is $\Sigma_{j_1,j_2} - L^{(t)}_{C_{j_1},C_{j_2}} + r_{j_1}^{(t)} r_{j_2}^{(t)} H^{(t)}_{C_{j_1},C_{j_2}} - \left(r_{j_1}^{(t)} + r_{j_2}^{(t)}\right) \Psi_{C_{j_1},C_{j_2}} + J^{(t)}_{C_{j_1},C_{j_2}} r_{j_1}^{(t)} + J^{(t)}_{C_{j_2},C_{j_1}} r_{j_2}^{(t)}$. If $C_{j_1} = C_{j_2} = c$, an additional term $\left(K_c^{(t)}\right)_{j_1,j_2}$ is subtracted.

12. We can now compute the scores for all $j \notin \mathcal{S}^{(t)}$ in $O\left(mn_M + \sum_{i=1}^{n_M} |M_i|^2\right)$. The overall complexity of the procedure described above is $O\left(mn_M + \sum_{i=1}^{n_M} |M_i|^2 \left|S^{(t)} \cap M_i\right|\right)$. The worst-case complexity for selecting all l probes is $O\left(mn_M + l^2 \sum_{i=1}^{n_M} |M_i|^2\right)$.

13. The imputation weights for probe j given $M_c \cap \mathcal{S}^{(t)}$ are: $\left(\Psi_{C_j,c} - J^{(t)}_{c,C_j} - r^{(t)}_j H^{(t)}_{C_j,c}\right) B^{(t)}_c$. If $C_j = c$, a term of $\left(D^{(t)}_c\right)_j$ is added.

2.6.3 Soft modular Gaussian

In this method for estimating the covariance matrix, we use the prior $J_{j_1,j_2} \sim \mathcal{N}\left(0; \eta^2_{j_1,j_2}\right)$ where $\eta^2_{j_1,j_2} = \frac{1}{2\gamma_1 \exp\left(-\gamma_2 r_{j_1,j_2}\right)}$ and r_{j_1,j_2} is the linear Pearson correlation between probes j_1 and j_2: $r_{j_1,j_2} = \frac{\sum_{i=1}^{n}\left(X_{j_1,i} - \bar{X}_{j_1}\right)\left(X_{j_2,i} - \bar{X}_{j_2}\right)}{\sqrt{\sum_{i=1}^{n}\left(X_{j_1,i} - \bar{X}_{j_1}\right)^2 \sum_{i=1}^{n}\left(X_{j_2,i} - \bar{X}_{j_2}\right)^2}}$. This gives rise to the penalty term $\gamma_1 \sum_{j_1=1}^{m}\sum_{j_2=1}^{m} \exp\left(-\gamma_2 r_{j_1,j_2}\right) J^2_{j_1,j_2}$.

We estimate the covariance matrix Σ using the method described in Section 2.5.5.

γ_1 and γ_2 are chosen using cross-validation in the following procedure: first, we choose γ by cross-validation to optimize the likelihood with the constant penalty $\gamma \sum_{j_1=1}^{m}\sum_{j_2=1}^{m} J^2_{j_1,j_2}$ which corresponds to the prior $J_{j_1,j_2} \sim \mathcal{N}\left(0; \frac{1}{2\gamma}\right)$. With this prior, the expected Frobenius norm of the precision matrix is $\mathbb{E}\left[\sum_{j_1=1}^{m}\sum_{j_2=1}^{m} J^2_{j_1,j_2}\right] = \frac{m^2}{2\gamma}$. Then, for each γ_2, we choose γ_1 to give the same expected Frobenius norm: $\sum_{j_1=1}^{m}\sum_{j_2=1}^{m} \frac{1}{2\gamma_1 \exp\left(-\gamma_2 r_{j_1,j_2}\right)} = \frac{m^2}{2\gamma}$. Hence, $\gamma_1 = \frac{\gamma \sum_{j_1=1}^{m}\sum_{j_2=1}^{m} \exp\left(\gamma_2 r_{j_1,j_2}\right)}{m^2}$. We choose γ_2 and the matching γ_1 using cross-validation.

2.7 Mixture models

2.7.1 Definition

Let $\{P_\theta(X) | \theta \in \Theta\}$ be a parametric family of distributions where Θ is the parameter space, $C \sim P(C|\phi)$ be a discrete class variable (ϕ parameterizes the class distribution) and $X|C = c \sim P_{\theta_c}$. This defines a mixture model over X with C being the class variable. Let $P(\theta)$ be a prior over θ. We assume that $(\theta_{c_1} \perp \theta_{c_2})$ for all $c_1 \neq c_2$. Then the marginal likelihood is:

$$P(\boldsymbol{X}) = \int_\phi P(\phi) \int_{\boldsymbol{\theta}} \left(\prod_c P(\theta_c)\right) \prod_{i=1}^{n} \sum_c P(C = c|\phi) P_{\theta_c}(X_i) d\boldsymbol{\theta} d\phi$$

2.7.2 Learning

We learn the model using the EM algorithm. In the E-step, we fix $\boldsymbol{\theta}$ and ϕ and compute $Q_i(C_i = c) = P(C_i = c|X_i, \boldsymbol{\theta}) = \frac{P_{\theta_c}(X_i) P(C_i = c|\phi)}{\sum_{c'} P_{\theta_c}(X_i) P(C_i = c|\phi)}$. In the M-step, we use the expected sufficient statistics $Q(C_i = c)$ to re-estimate $\boldsymbol{\theta}$ and ϕ by maximizing the expected log-likelihood with respect to Q:

$$\langle \hat{\boldsymbol{\theta}}, \hat{\phi} \rangle \quad = \quad \arg\max_{\langle \boldsymbol{\theta}, \phi \rangle} \log P(\phi) + \sum_c \log P(\theta_c) + \sum_{i=1}^{n} \sum_c Q_i(c) \left(\log P(C = c|\phi) + \log P_{\theta_c}(X_i) \right)$$

Since we assume parameter independence, we can solve separately:

$$\begin{aligned} \hat{\phi} &= \arg\max_{\phi} \log P(\phi) + \sum_{i=1}^{n} \sum_c Q_i(c) \log P(C = c|\phi) \\ \hat{\theta}_c &= \arg\max_{\theta_c} \log P(\theta_c) + \sum_{i=1}^{n} Q_i(c) \log P_{\theta_c}(X_i) \end{aligned}$$

2.7.3 Selection

Given a probeset $\mathcal{S}$ and a data sample X_i, the inferred distribution given $X_{\mathcal{S},i}$ is:

$$Q_i \left(C_i, X_i | X_{\mathcal{S},i} \right) \quad = \quad P \left(C_i | X_{\mathcal{S},i} \right) P \left(X_{-\mathcal{S},i} | C_i, X_{\mathcal{S},i} \right)$$

The KL divergence between the learned mixture-model distribution $Q_i \left(C_i, X_i \right) = Q_i \left(C_i \right) P_{\theta_{C_i}} \left(X_i \right)$ and the inferred distribution $Q \left(C_i, X_i | X_{\mathcal{S},i} \right)$ is:

$$\begin{aligned} D_i \left[\mathcal{S} \right] &= KL \left(Q_i \left(C_i, X_i \right) \| Q \left(C_i, X_i | X_{\mathcal{S},i} \right) \right) \\ &= \text{const} - \sum_c Q_i(c) \log Q(c|X_{\mathcal{S},i}) - \sum_c Q_i(c) \log P_{\theta_c} \left(X_{-\mathcal{S},i} | X_{\mathcal{S},i} \right) \\ &= \text{const} + \sum_c Q_i(c) \log P_{\theta_c} \left(X_{\mathcal{S},i} \right) - \sum_c Q_i(c) \log Q \left(c | X_{\mathcal{S},i} \right) \\ Q(c|X_{\mathcal{S},i}) &= \frac{Q(c) P_{\theta_c}(X_{\mathcal{S},i})}{\sum_{c'} Q(c') P_{\theta_{c'}}(X_{\mathcal{S},i})} \end{aligned}$$

To minimize this distance, we would like to choose a probeset $\mathcal{S}$ to minimize $\sum_{i=1}^{n} D_i \left[\mathcal{S} \right]$. Using a greedy algorithm, this only requires us to compute at each iteration $P_{\theta_c} \left(X_{\mathcal{S} \cup \{j\},i} \right)$ for all c and j.

2.7.4 Selection for mixtures of Gaussians

If $\theta_c = \langle \mu_c, \Sigma_c \rangle$ and $X | \theta_c \sim \mathcal{N} \left(\mu_c; \Sigma_c \right)$, then

$$\begin{aligned} P_{\theta_c} \left(X_{\mathcal{S},i} \right) &= (2\pi)^{-|\mathcal{S}|/2} \left| (\Sigma_c)_{\mathcal{S},\mathcal{S}} \right|^{-1/2} \exp \left(-\frac{1}{2} \left(X_{\mathcal{S},i} - (\mu_c)_{\mathcal{S}} \right)^T (\Sigma_c)^{-1}_{\mathcal{S},\mathcal{S}} \left(X_{\mathcal{S},i} - (\mu_c)_{\mathcal{S}} \right) \right) \\ &= \log P_{\theta_c} \left(X_i \right) - \log P_{\theta_c} \left(X_{\mathcal{S},i} \right) \end{aligned}$$

Let $C = \Sigma_c$, then to compute $C^{-1}_{\mathcal{S} \cup \{j\}, \mathcal{S} \cup \{j\}}$ efficiently using $C^{-1}_{\mathcal{S},\mathcal{S}}$ we use:

$$C^{-1}_{\mathcal{S} \cup \{j\}, \mathcal{S} \cup \{j\}} = \begin{pmatrix} \frac{C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j} C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}}}{C_{j,j} - C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j}} + C^{-1}_{\mathcal{S},\mathcal{S}} & \frac{C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j}}{C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j} - C_{j,j}} \\ \frac{C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}}}{C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j} - C_{j,j}} & \frac{1}{C_{j,j} - C_{j,\mathcal{S}} C^{-1}_{\mathcal{S},\mathcal{S}} C_{\mathcal{S},j}} \end{pmatrix}$$

Now, let $x = (X_{\mathcal{S},i} - \mu_{\mathcal{S}})$, $y = x_{j,i} - \mu_j$, $v = C_{\mathcal{S},\mathcal{S}}^{-1}(X_{\mathcal{S},i} - \mu_{\mathcal{S}})$, $u = C_{j,\mathcal{S}}v$, $d = \frac{1}{C_{j,j} - C_{j,\mathcal{S}}C_{\mathcal{S},\mathcal{S}}^{-1}C_{\mathcal{S},j}}$, then

$$\left(X_{\mathcal{S}\cup\{j\}} - \mu_{\mathcal{S}\cup\{j\}}\right)^T C_{\mathcal{S}\cup\{j\},\mathcal{S}\cup\{j\}}^{-1}\left(X_{\mathcal{S}\cup\{j\}} - \mu_{\mathcal{S}\cup\{j\}}\right) = d(u-y)^2 + x^T C_{\mathcal{S},\mathcal{S}}^{-1} x$$

To compute $|C_{\mathcal{S}\cup\{j\}}|$ we use:

$$\begin{pmatrix} C_{\mathcal{S},\mathcal{S}}^{-1} & 0 \\ 0 & 1 \end{pmatrix} C_{\mathcal{S}\cup\{j\},\mathcal{S}\cup\{j\}} = \begin{pmatrix} I & C_{\mathcal{S},\mathcal{S}}^{-1}C_{\mathcal{S},j} \\ C_{j,\mathcal{S}} & C_{j,j} \end{pmatrix}$$
$$\left|C_{\mathcal{S}\cup\{j\},\mathcal{S}\cup\{j\}}\right| = |C_{\mathcal{S},\mathcal{S}}|\left(C_{j,j} - C_{j,\mathcal{S}}C_{\mathcal{S},\mathcal{S}}^{-1}C_{\mathcal{S},j}\right)$$

Hence,

$$-2\log\frac{P_{\theta_c}\left(X_{\mathcal{S}\cup\{j\},i}\right)}{P_{\theta_c}\left(X_{\mathcal{S},i}\right)} = \log(2\pi) + \log\left(C_{j,j} - C_{j,\mathcal{S}}C_{\mathcal{S},\mathcal{S}}^{-1}C_{\mathcal{S},j}\right) + \frac{\left(C_{j,\mathcal{S}}C_{\mathcal{S},\mathcal{S}}^{-1}(X_{\mathcal{S},i} - \mu_{\mathcal{S}}) - (x_{j,i} - \mu_j)\right)^2}{C_{j,j} - C_{j,\mathcal{S}}C_{\mathcal{S},\mathcal{S}}^{-1}C_{\mathcal{S},j}}$$

2.7.5 Multinomial class prior

Let $P(C = c|\phi) = \phi_c$ where $1 \leq c \leq n_c$ and $\sum_{c=1}^{n_c}\phi_c = 1$. Also let $\phi \sim \text{Dirichlet}\left(\frac{\alpha}{n_c}, \ldots, \frac{\alpha}{n_c}\right)$ be a BDe prior over ϕ corresponding to a uniform distribution and an imaginary sample size of α. Denote the expected sufficient statistics by $\bar{M}[c] = \sum_{i=1}^{n} Q_i(c)$ and $\bar{M} = n$. Then the posterior over ϕ using the expected sufficient statistics is Dirichlet $\left(\frac{\alpha}{n_c} + \bar{M}[1], \ldots, \frac{\alpha}{n_c} + \bar{M}[n_c]\right)$. Taking the expected value then gives $\phi_c = \frac{\frac{\alpha}{n_c} + \bar{M}[c]}{\alpha + n}$.

On the other hand, to use the MAP solution in the M-step, we introduce the Lagrange multiplier η for the constraint $\sum_c \phi_c = 1$ and take partial derivatives:

$$\begin{aligned}
\mathcal{L}(\phi, \eta) &= \sum_{c=1}^{n_c}\left(\frac{\alpha}{n_c} - 1\right)\log(\phi_c) + \sum_{c=1}^{n_c}\log(\phi_c)\left(\sum_{i=1}^{n} Q_i(c)\right) + \eta\left(\sum_{c=1}^{n_c}\phi_c - 1\right) \\
\frac{\partial L}{\partial \phi_c} &= \frac{1}{\phi_c}\left(\frac{\alpha}{n_c} - 1 + \sum_{i=1}^{n} Q_i(c)\right) + \eta \\
\phi_c &= \frac{-1}{\eta}\left(\frac{\alpha}{n_c} - 1 + \sum_{i=1}^{n} Q_i(c)\right) \\
1 &= \frac{-1}{\eta}(n + \alpha - n_c) \Rightarrow \eta = -(n + \alpha - n_c) \\
\phi_c &= \frac{1}{n + \alpha - n_c}\left(\frac{\alpha}{n_c} - 1 + \sum_{i=1}^{n} Q_i(c)\right) = \frac{\frac{\alpha}{n_c} - 1 + \bar{M}[c]}{\alpha + n - n_c}
\end{aligned}$$

To choose the number of clusters, we can use cross-validation.

2.7.6 Mixture of Gaussians

We use $\theta_c = \langle \mu_c, \Sigma_c \rangle$ and $X|\theta_c \sim \mathcal{N}(\mu_c; \Sigma_c)$. Assume the prior on μ_c is $\mu_c \sim \mathcal{N}(\mu, \Sigma)$. Then the maximum likelihood estimate for μ_c is:

$$\begin{aligned}
l_Q(\mu_c) &= \log P(\mu_c) + \sum_{i=1}^{n} Q_i(c) \log P_{\theta_c}(X_i) \\
&= \text{const} - \frac{1}{2}(\mu_c - \mu)^T \Sigma^{-1}(\mu_c - \mu) - \frac{1}{2}\sum_{i=1}^{n} Q_i(c)(X_i - \mu_c)^T \Sigma_c^{-1}(X_i - \mu_c) \\
0 &= \Sigma^{-1}(\mu_c - \mu) + \Sigma_c^{-1}\sum_{i=1}^{n} Q_i(c)(\mu_c - X_i) \\
\mu_c &= \left(\Sigma^{-1} + \Sigma_c^{-1}\sum_{i=1}^{n} Q_i(c)\right)^{-1}\left(\Sigma^{-1}\mu + \Sigma_c^{-1}\sum_{i=1}^{n} Q_i(c)X_i\right)
\end{aligned}$$

If we don't use a prior on μ_c, the maximum likelihood estimate is simply:

$$\mu_c = \frac{\sum_{i=1}^{n} Q_i(c)X_i}{\sum_{i=1}^{n} Q_i(c)}$$

In general, we can use any of the Gaussian models described above for Σ_c. Following are some specific cases. We need to estimate a covariance matrix using one of the methods described above, but using weights on the data. We denote by w_i the individual weights, by $W \in \mathbb{R}^{1\times n}$ the row vector containing the weights and by $\boldsymbol{W} \in \mathbb{R}^{n\times n}$ the diagonal matrix with W on its diagonal.

2.7.7 Mixture of L_2-regularized estimated precision matrices

In this formulation, we place a prior on the estimated covariance matrix $\hat{\Sigma}$, which is a Gaussian prior on the entries of the precision matrix Σ^{-1}: $P(\Sigma) = (2\pi)^{\frac{-m^2}{2}} \sigma^{-m^2} \exp\left(\frac{-1}{2\sigma^2}\sum_{i=1}^{m}\sum_{j=1}^{m}\left(\Sigma_{i,j}^{-1}\right)^2\right)$. Then, the weighted likelihood is:

$$\begin{aligned}
-2l(\Sigma : \boldsymbol{X}) &= mn\log(2\pi) - \sum_{i=1}^{n} w_i \log\left|\Sigma^{-1}\right| + \sum_{i=1}^{n} w_i \left(X_i - \bar{X}\right)^T \Sigma^{-1}\left(X_i - \bar{X}\right) + \\
&+ m\log(2\pi) + 2m^2\log\sigma + \frac{1}{\sigma^2}\operatorname{Tr}\left[\Sigma^{-2}\right] \\
\frac{\partial(-2l(\Sigma : \boldsymbol{X}))}{\partial\Sigma^{-1}} &= -\sum_{i=1}^{n} w_i\Sigma + \sum_{i=1}^{n} w_i\left(X_i - \bar{X}\right)\left(X_i - \bar{X}\right)^T + \frac{2}{\sigma^2}\Sigma^{-1}
\end{aligned}$$

Let $\sum_{i=1}^{n} w_i\left(X_i - \bar{X}\right)\left(X_i - \bar{X}\right)^T = U\Lambda U^T$ be the eigen-decomposition of the weighted-data empirical covariance matrix. Setting the gradient to 0:

$$
\begin{aligned}
0 &= \left(\sum_{i=1}^{n} w_i\right)\Sigma^2 - \Sigma\sum_{i=1}^{n} w_i\left(X_i-\bar{X}\right)\left(X_i-\bar{X}\right)^T - \frac{2}{\sigma^2}I \\
\Sigma &= \frac{\sum_{i=1}^{n} w_i\left(X_i-\bar{X}\right)\left(X_i-\bar{X}\right)^T \pm \left(\left(\sum_{i=1}^{n} w_i\left(X_i-\bar{X}\right)\left(X_i-\bar{X}\right)^T\right)^2 + \frac{8\sum_{i=1}^{n} w_i}{\sigma^2}I\right)^{1/2}}{2\sum_{i=1}^{n} w_i} \\
&= \frac{1}{2\sum_{i=1}^{n} w_i}\left(U\Lambda U^T \pm \left(U\left(\Lambda^2 + \frac{8\sum_{i=1}^{n} w_i}{\sigma^2}I\right)U^T\right)^{1/2}\right) \\
&= \frac{1}{2\sum_{i=1}^{n} w_i}U\left(\Lambda + \sqrt{\Lambda^2 + \frac{8\sum_{i=1}^{n} w_i}{\sigma^2}I}\right)U^T
\end{aligned}
$$

The minus solution results in a matrix which is not positive semi-definite, so the only solution is the plus solution.

References

[1] J. Duchi, S. Gould, and D. Koller. Projected subgradient methods for learning sparse gaussians. In *Proceedings of the Twenty-fourth Conference on Uncertainty in AI (UAI)*, 2008.

[2] D Oleary and G Stewart. Computing the eigenvalues and eigenvectors of symmetric arrowhead matrices? *Journal of Computational Physics*, 90(2):497–505, 1990.

[3] Juliane Schafer and Korbinian Strimmer. A shrinkage approach to Large-Scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, 4(1), 2005.