Unsupervised Topic Modelling of Cell Type Mixtures in Spatial Transcriptomics Data

Michael Diao

Rafalab Meeting 8/14

Thanks!

- Dylan
- Professor Chen (Broad Institute)
- Professor Irizarry
- You guys!

Why?

- RCTD provides a supervised approach to estimating cell type weights.
- Want a way to estimate cell type weights *without* reference data.

Approaches, Progress & Setbacks

Approach 0: Latent Dirichlet Allocation

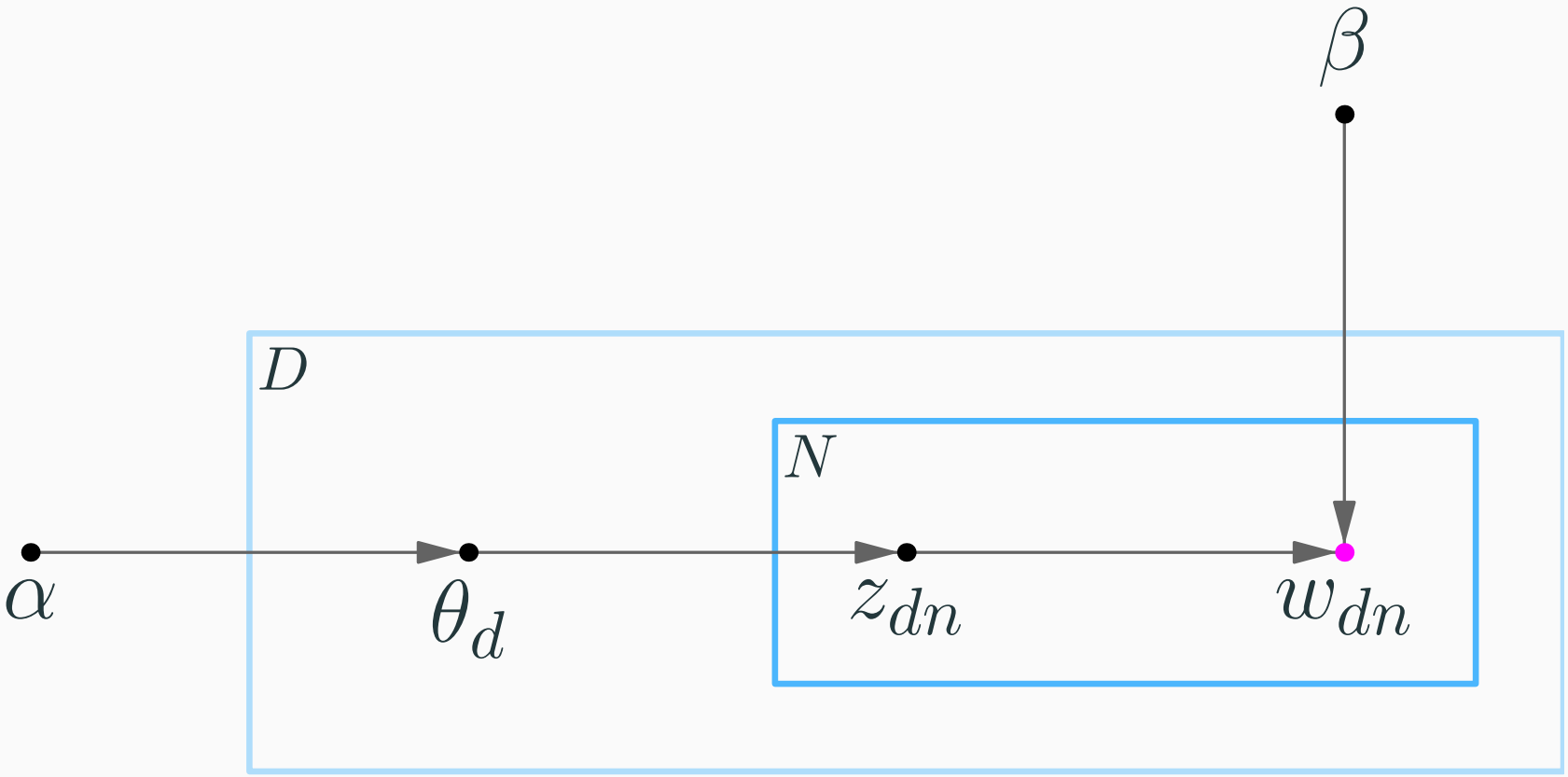


Figure 1: Hierarchical model for LDA.

Approach 0: Attempting to use out-of-box LDA

```
from sklearn.decomposition import
 ↪  LatentDirichletAllocation as LDA
lda = LDA(n_components=3, max_iter=100, random_state=0)
lda.fit(doublets)
print(lda.transform(doublets))
```

Purkinje?	Astrocytes?	Granule?
0.1745784	0.17457841	0.65084319
0.17622564	0.17622564	0.64754872
0.17392182	0.17392182	0.65215635
⋮	⋮	⋮
0.17503798	0.17503798	0.64992403
0.17510081	0.17510081	0.64979839
0.17310268	0.17310268	0.65379464

Table 1: What's going on?

Approach 0: Pros/Cons

- Pros:
  - Lots of literature on LDA—very popular topic model
  - Fast due to variational approximation
- Cons:
  - Hard to interpret/debug
  - Model not necessarily suited for our particular problem
  - Doesn't incorporate sparsity information, gene overdispersion

Approach 1: Bayesian MCMC with *PyStan*

```
import pystan
poisson_code = """
data {
    int<lower=0> N[P,J];
    ...
}
parameters {
    simplex[types] theta[P];
    simplex[J] beta[types];
}
model {
    for (p in 1:P)
        N[p] ~ poisson(I[p] * theta[p]' * beta[types]);
}
"""
poisson_model =
 ↪  pystan.StanModel(model_code=poisson_code)
```

Approach 1: Bayesian MCMC with *PyStan*

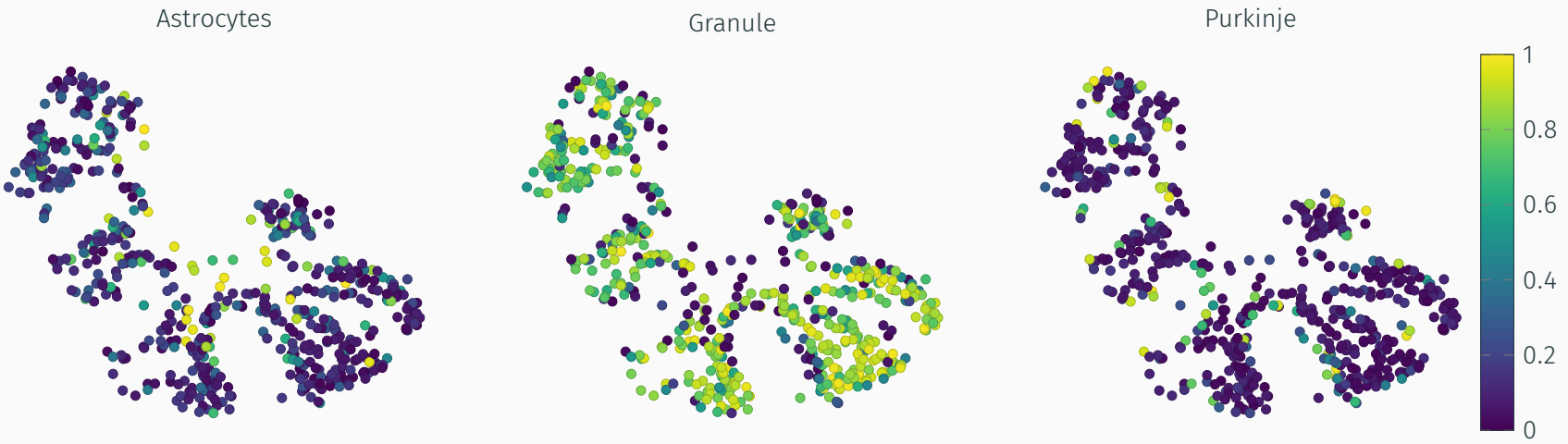


Figure 2: Testing unsupervised PyStan MCMC approach with Poisson model.

Approach 1: Bayesian MCMC with *PyStan*

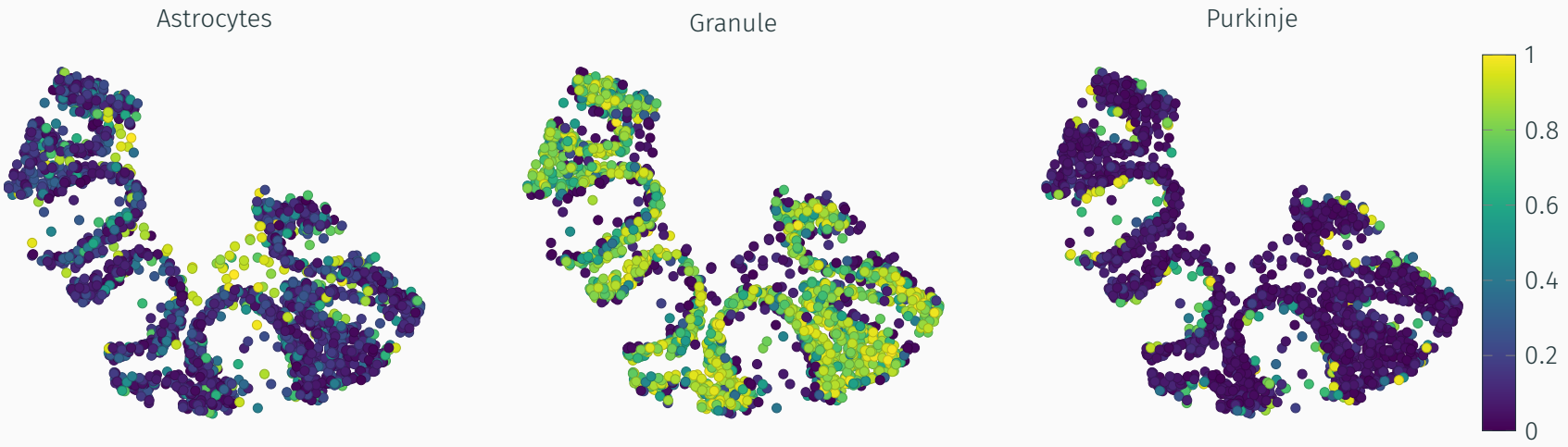


Figure 3: Testing MCMC on many cells.

An Issue

Purkinje	Astrocytes	Granule
0.233	0.674	0.092
0.063	0.098	0.838
0.197	0.085	0.717
0.071	0.412	0.515
0.026	0.041	0.932
0.022	0.957	0.020
0.051	0.120	0.828
0.021	0.173	0.804
0.043	0.414	0.541
0.059	0.142	0.797

Table 2: “Astrocyte pollution.”

An Issue

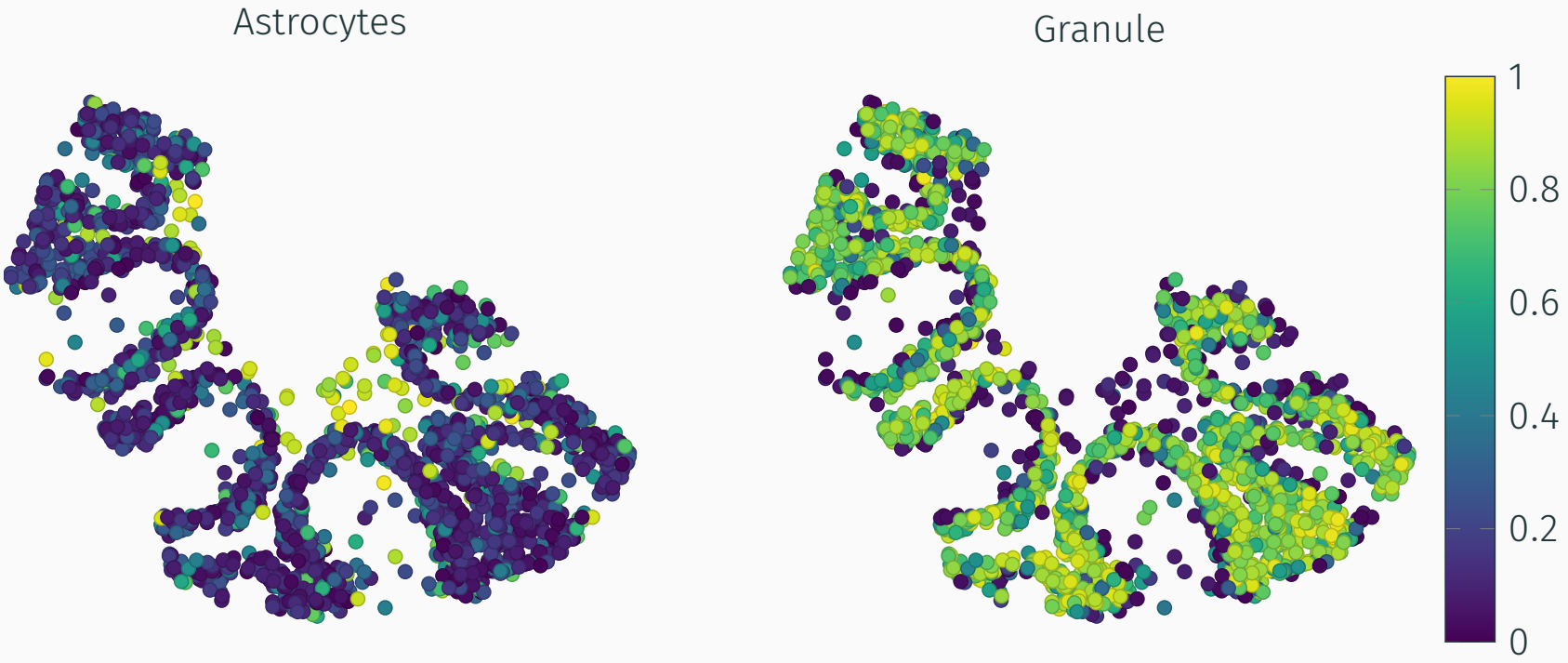


Figure 4: “Astrocyte pollution.”

Attempted Remedy: Introducing a Prior

Dirichlet distribution

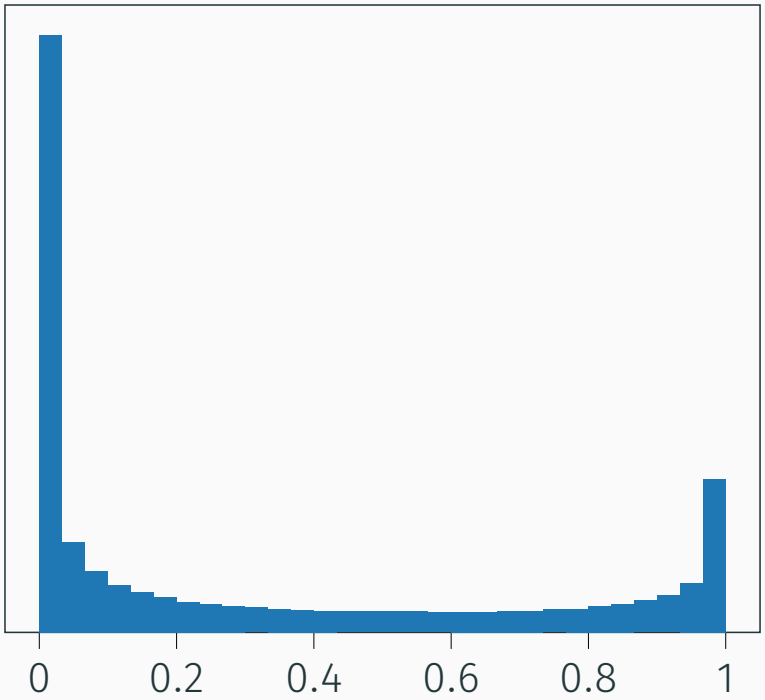
$$\mathbb{P}(W \mid \alpha) \propto \prod_{i=1}^{K} W_i^{\alpha_i - 1}$$

Attempted Remedy: Introducing a Prior

Dirichlet distribution

$$\mathbb{P}(W \mid \alpha) \propto \prod_{i=1}^{K} W_i^{\alpha_i - 1}$$

Relative frequencies of weights for $\alpha = \langle 0.2, 0.2, 0.2 \rangle$

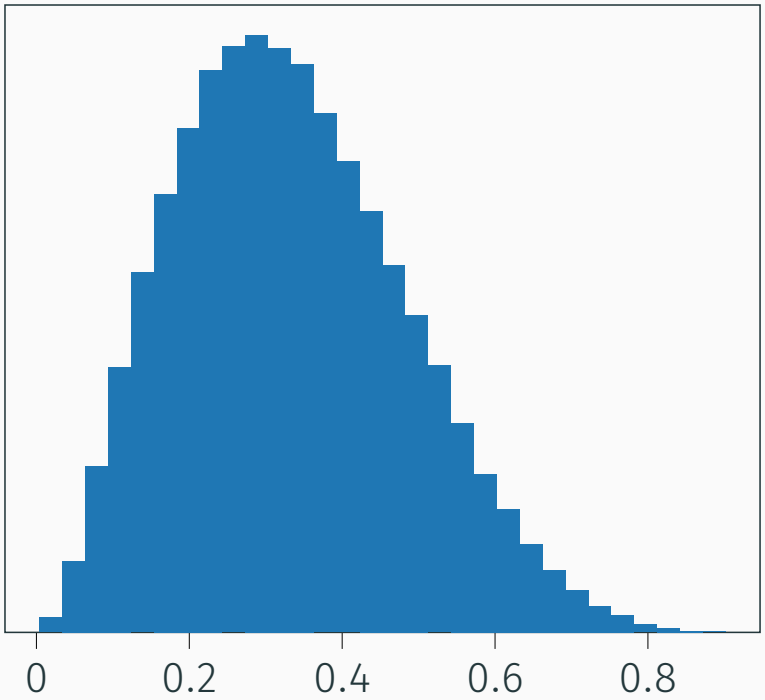


Attempted Remedy: Introducing a Prior

Dirichlet distribution

$$\mathbb{P}(W \mid \alpha) \propto \prod_{i=1}^{K} W_i^{\alpha_i - 1}$$

Relative frequencies of weights for $\alpha = \langle 2, 2, 2 \rangle$



Approach 1: Pros/Cons

- Pros:
  - Very accurate results for simulated data
  - Results for real data look good
  - Initialization doesn't really seem to matter
- Cons:
  - Fully Bayesian MCMC is extremely slow
  - Took a week for results on 7 cell types!

Approach 2: Variational Bayes (ADVI) with *PyStan*

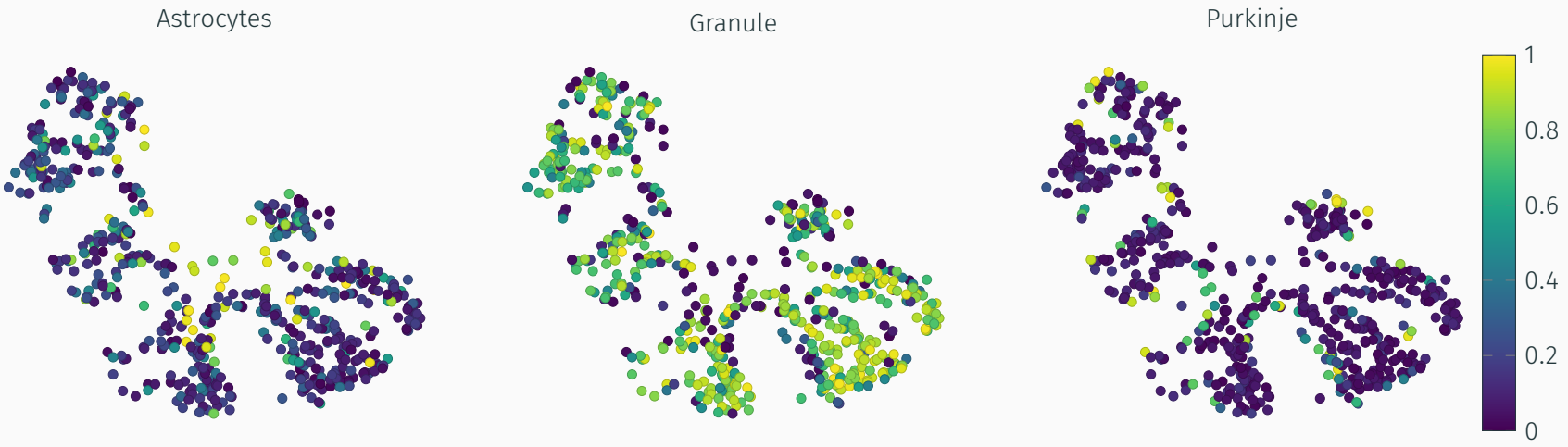


Figure 5: Testing Variational Bayes.

Approach 2: Variational Bayes (ADVI) with *PyStan*

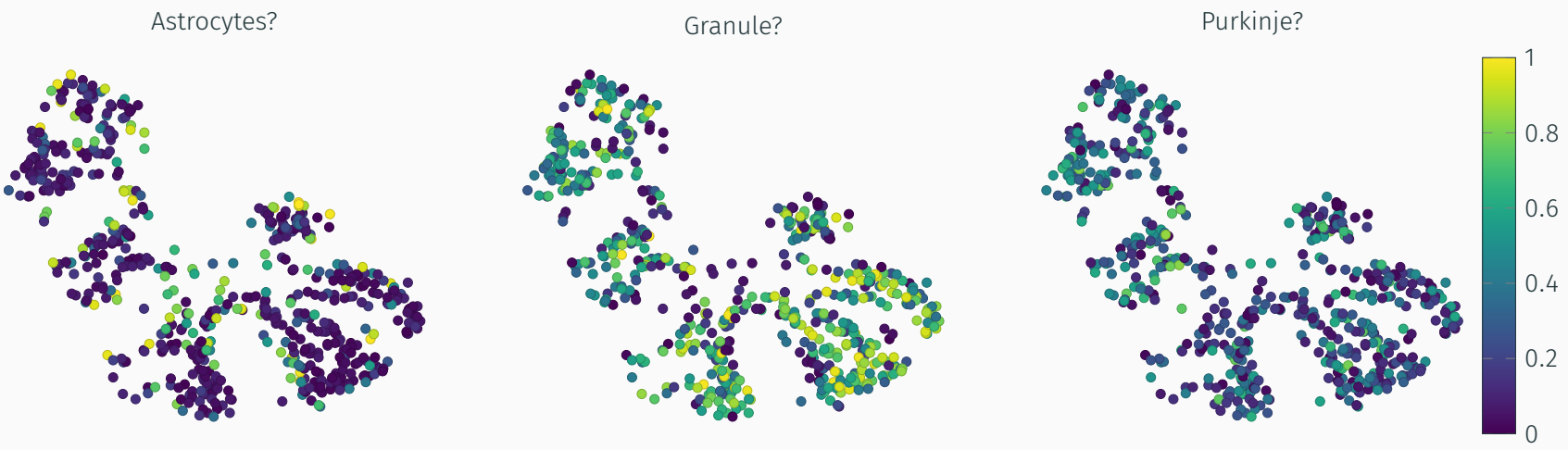


Figure 6: Bad initialization!

Approach 2: Variational Bayes (ADVI) with *PyStan*

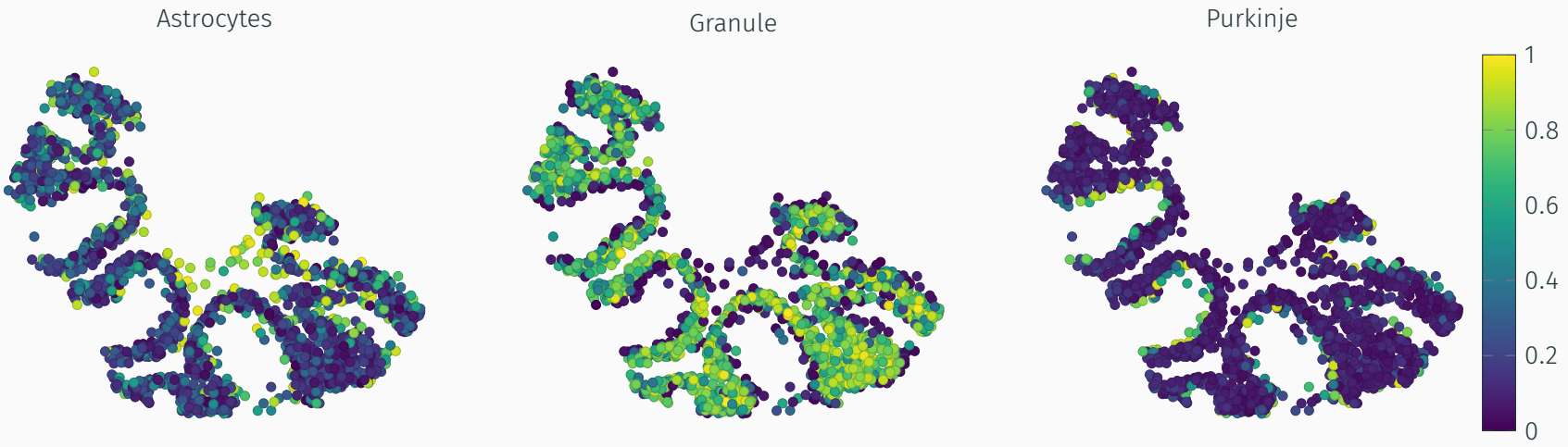


Figure 7: More astrocyte pollution!

Approach 2: Variational Bayes (ADVI) with *PyStan*

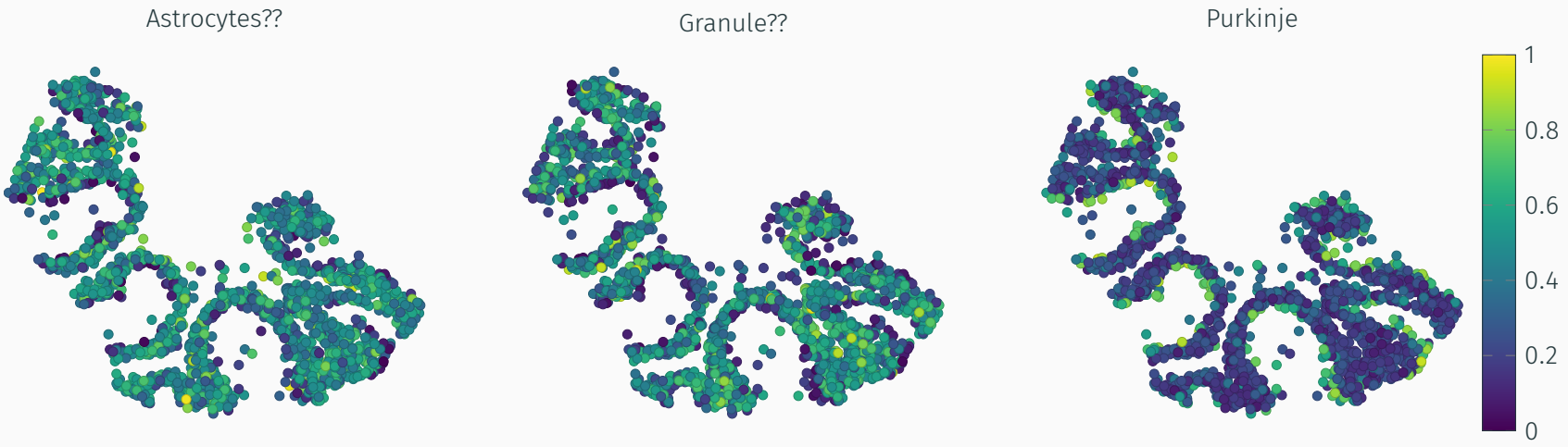


Figure 8: No longer just “pollution”...

Approach 2: Pros/Cons

- Pros
  - Much faster than MCMC
  - At least 10× speedup on 6311 pixel, 3 type dataset
- Cons
  - Strong independence assumptions
  - Prone to getting suck at local minima
  - Cell type mixing, again—but worse
  - Hard to debug

Where's the prior?

```
import pystan
dir_prior_code = """..."""
dir_prior_model =
  ↪  pystan.StanModel(model_code=dir_prior_code)
vb_samples = dir_prior_model.vb(data = {...})
```

RuntimeError

```
stan::variational::normal_meanfield::calc_grad: The
number of dropped evaluations has reached its maximum
amount (10). Your model may be either severely
ill-conditioned or misspecified.
```

Back to square one

Approach 3: Maximum Likelihood

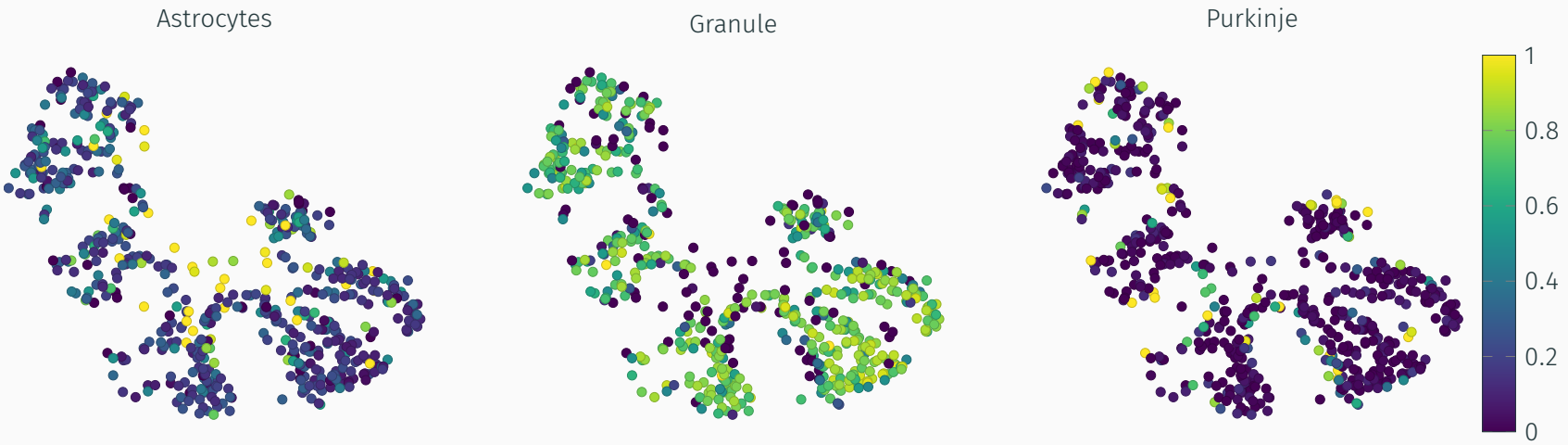


Figure 9: Testing unsupervised MLE approach: Poisson lognormal model, no sparsity prior

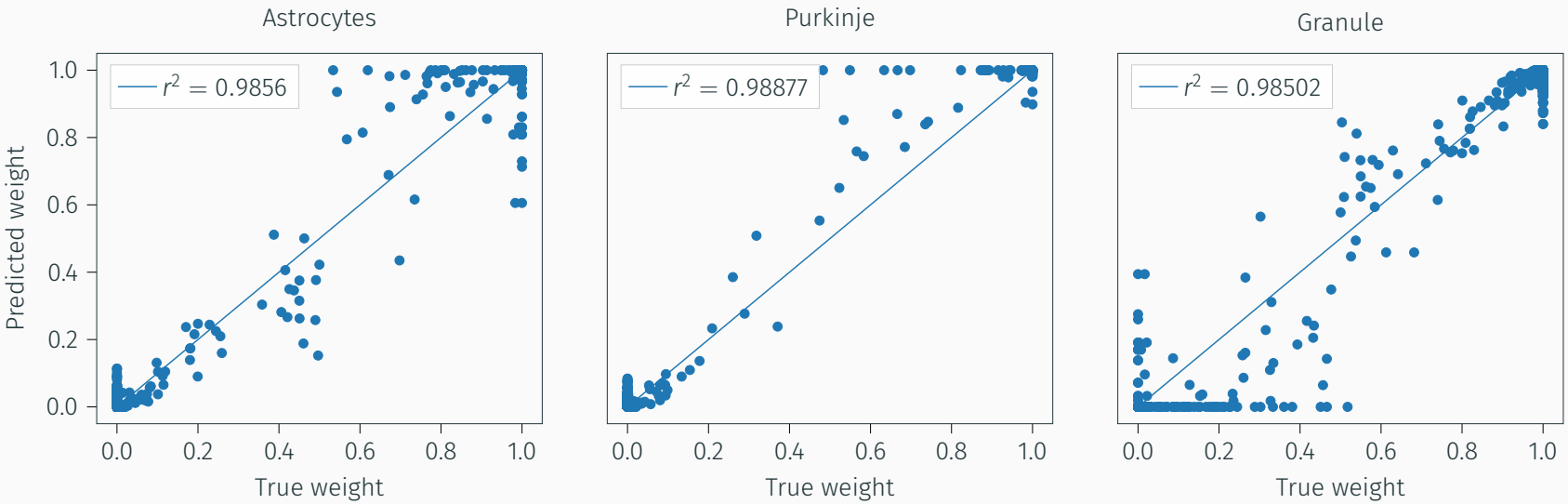


Figure 10: Testing IRWLS: retrieving cell type weights for 3 cells in fake data given true gene expression matrix.

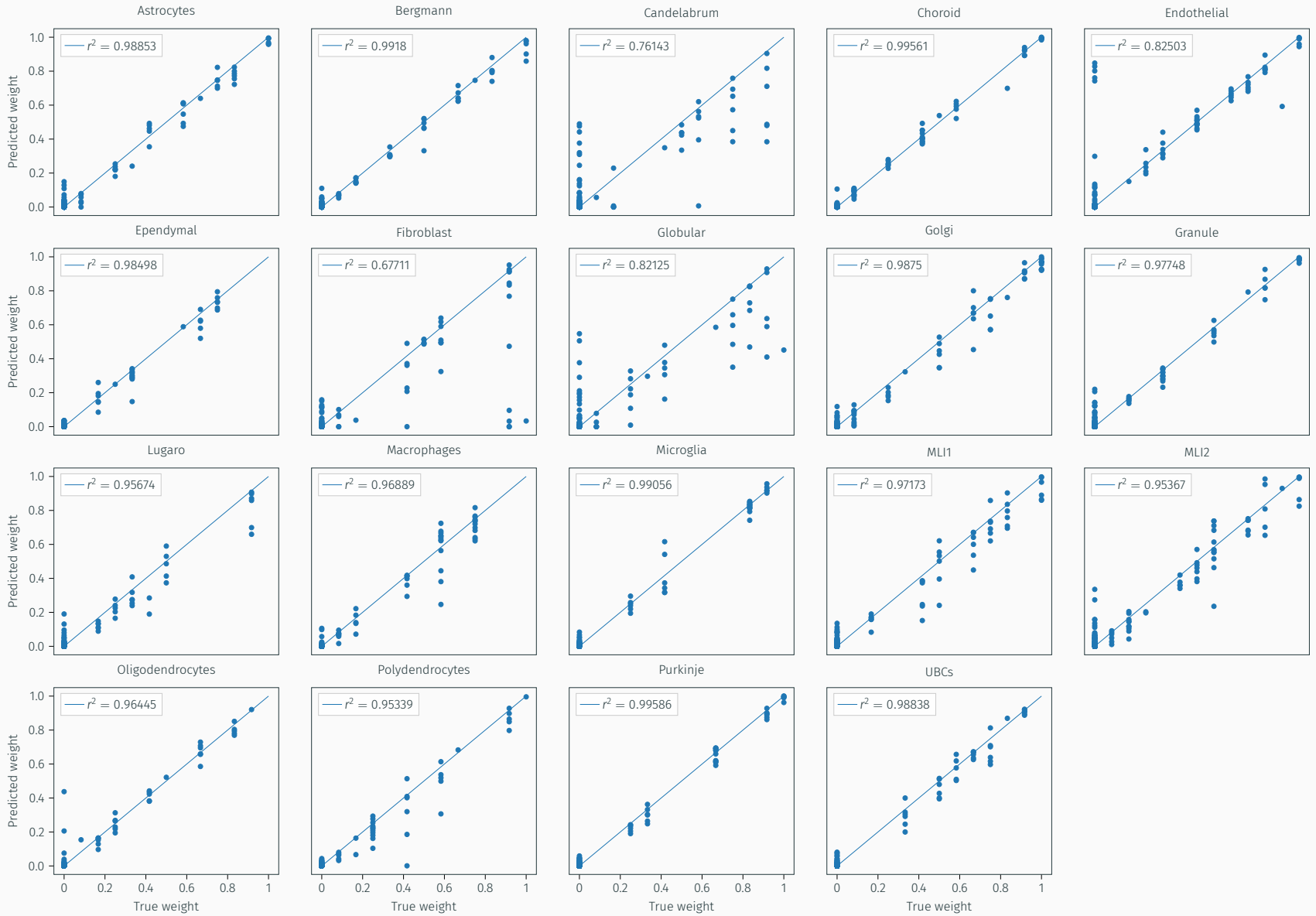


Figure 11: More testing: 19 cell types.

Approach 3: MLE $\rightarrow$ MAP Estimate

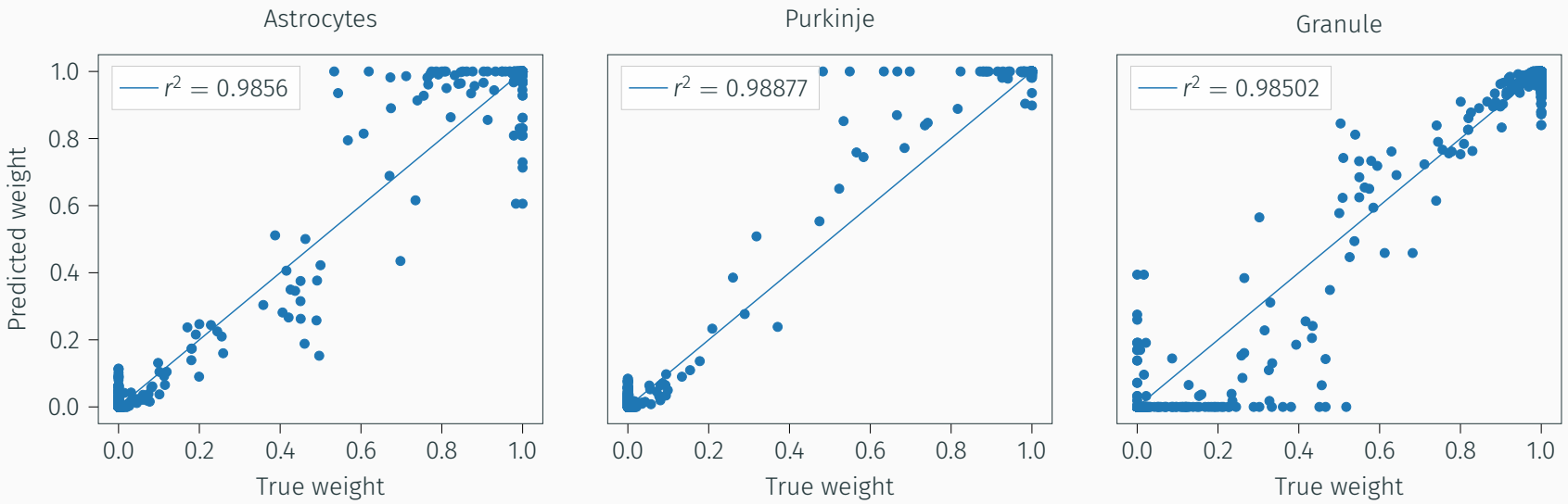


Figure 12: No prior.

Approach 3: MLE $\rightarrow$ MAP Estimate

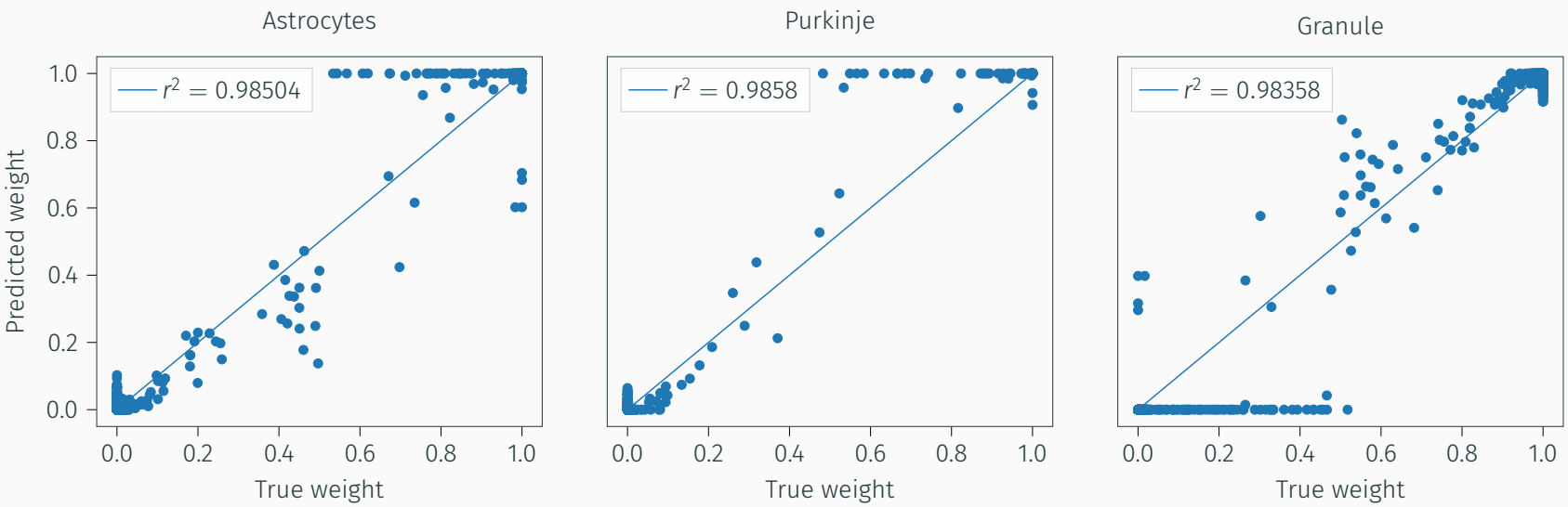


Figure 13: With Dirichlet prior, $\alpha = \langle 0.1, 0.1, 0.1 \rangle$.

Approach 3: MLE $\rightarrow$ MAP Estimate

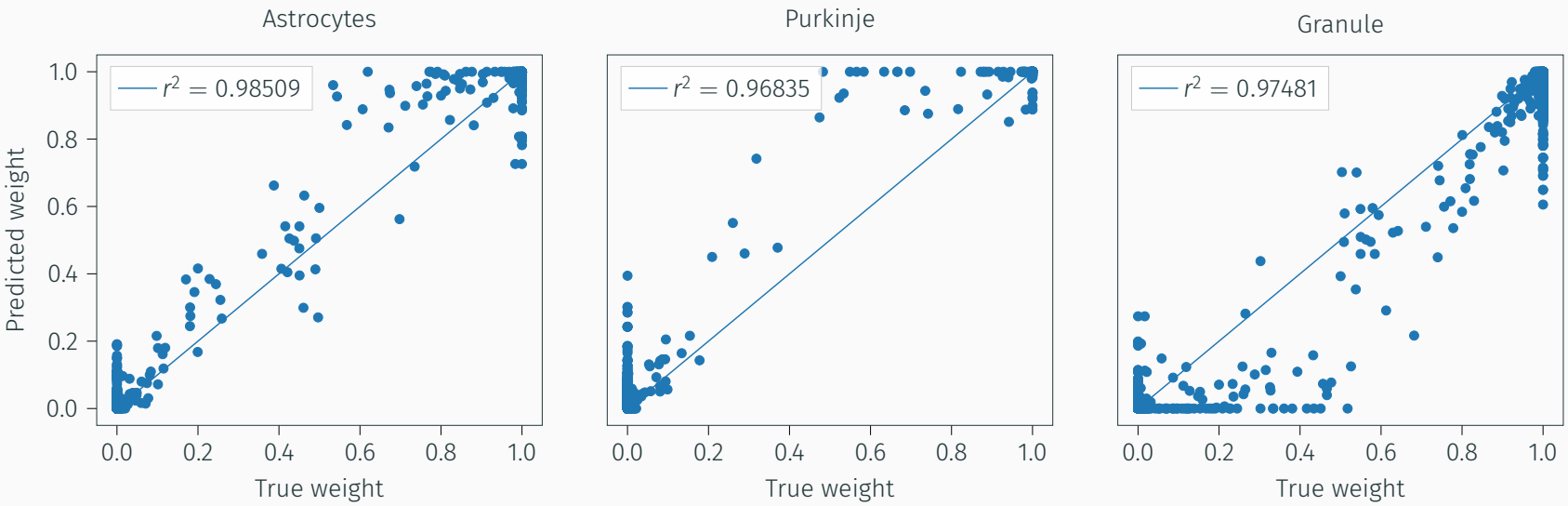


Figure 14: IRWLS altmax algorithm with snRNA init, no prior.

Approach 3: MLE $\rightarrow$ MAP Estimate

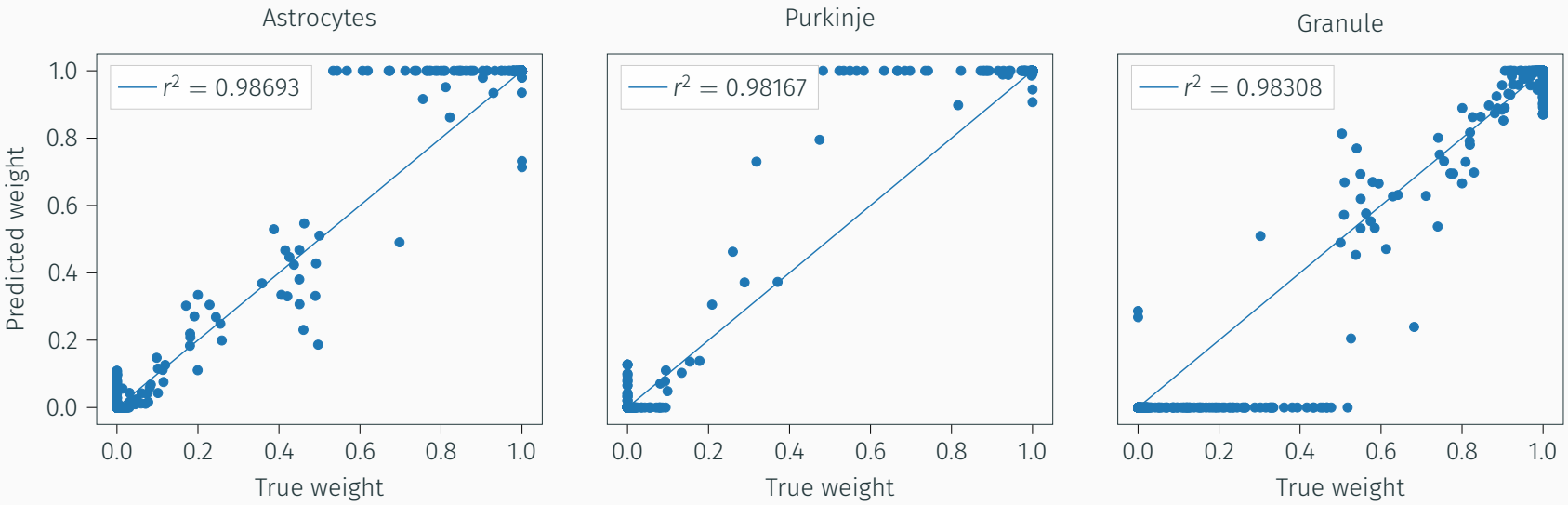


Figure 15: With Dirichlet prior, $\alpha = \langle 0.1, 0.1, 0.1 \rangle$.

Approach 3: Pros/Cons

- Pros
  - Implemented on top of RCTD's preexisting functions
  - Simpler to debug
  - Very fast
  - We can actually incorporate the prior!
- Cons
  - Right now, need to initialize with cross-reference—not unsupervised!
  - Initialization is hard...

```
solution += alpha * quadprog.solve_qp(...)
```

ValueError

```
Buffer dtype mismatch, expected 'double' but got
'complex double'
```

What now?

- How to increase sparsity?
- Multithreading in Python?

```
weights = [compute_weights(x) for x in something]
```

- Can we find a reasonable initialization without using a reference?
- Other approaches?
- Reach out to me at diao@mit.edu!

Thanks for listening!