Vertex finding by sparse model-based clustering

Korbinian Eckstein[1], **Rudi Frühwirth**[1]
Sylvia Frühwirth-Schnatter[2]

[1] *Institute of High Energy Physics*
Austrian Academy of Sciences, Vienna

[2] *Institute for Statistics and Mathematics*
Vienna University of Economics and Business

ACAT 2016, UTFSM, Valparaíso
January 21, 2016

Outline

Introduction

- Primary vertex finding can be interpreted as **1d clustering problem**
- We will compare two methods:

  Sparse model-based clustering

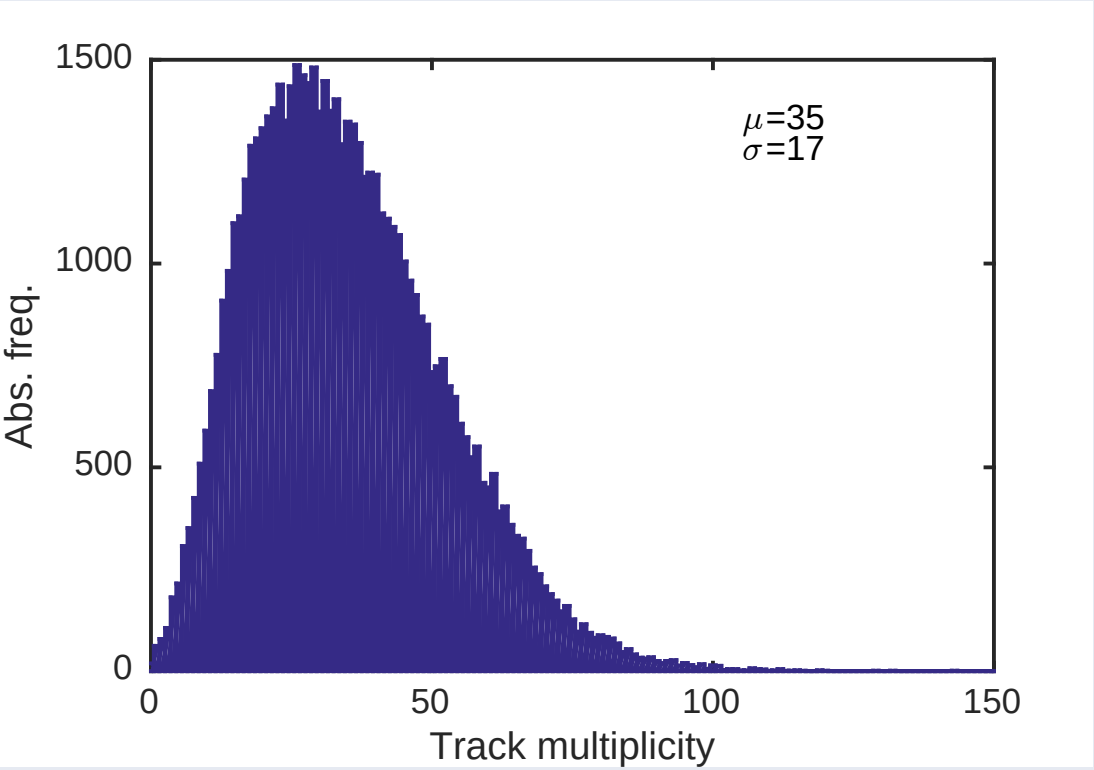
  and

  EM algorithm

- Model-based clustering can include available information as **prior densities**:
  - Number of clusters/vertices
  - Cluster size/number of tracks per vertex
  - Cluster/vertex spread
- Use a **normal model** for each cluster
- Study **performance** and **sensitivity to priors** with a simplified simulation

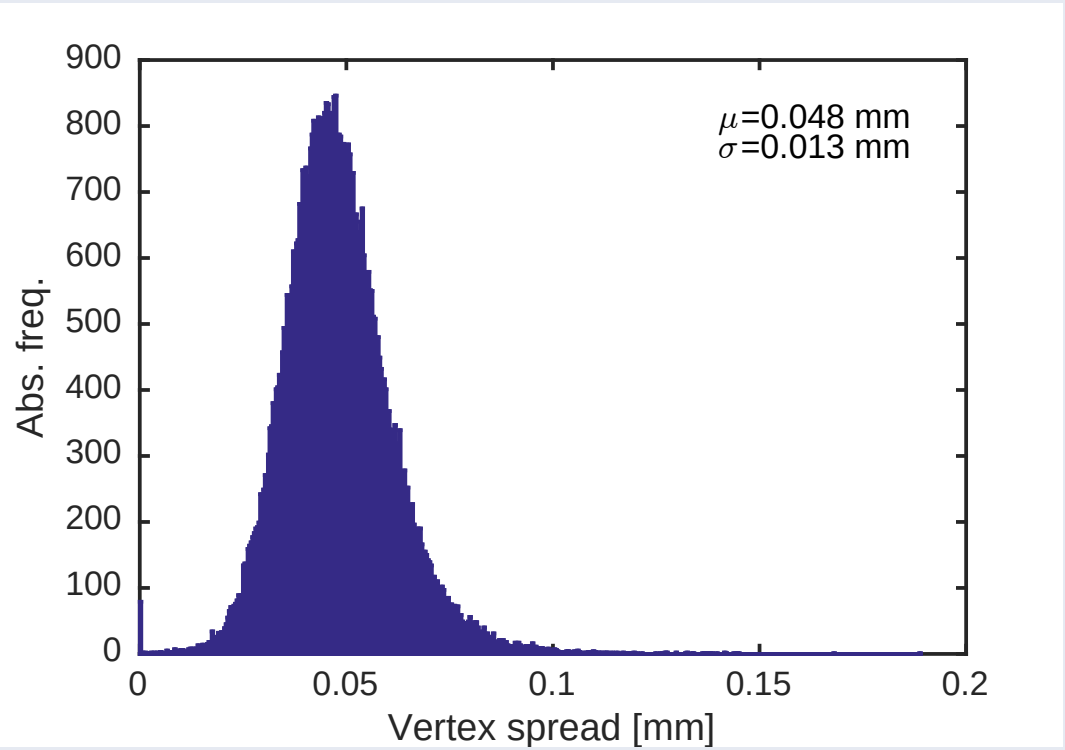
Track multiplicity

- We have simulated proton-proton interactions at LHC energy with PYTHIA and applied some basic cuts in p and η
- The empirical distribution $g(M)$ of the track multiplicity M per interaction vertex is smoothed by a kernel estimator and stored for further use



Vertex spread

- The z-positions of the tracks produced in an interaction are smeared by the extrapolation error from the innermost pixel layer and multiple scattering in the beam tube
- The empirical distribution of the resulting vertex spread s_k is described by its mean $\mu_s = 0.048\,\text{mm}$ and its standard deviation $\sigma_s = 0.013\,\text{mm}$



The Data

Bunch crossings

- A bunch-crossing consists of K superimposed interactions
- The number K is drawn from a Poisson distribution
- Each bunch crossing can be segmented into sections
- Cluster finding proceeds independently in each section

Number of components

- Assume that there are N tracks in a segment
- The three most likely numbers of clusters K_1, K_2, K_3 are obtained by looking up the likelihood of the multiplicity $M = N/K$ in the empirical distribution $g(M)$

Model and priors I

- **Input:** N tracks with z-positions $z_i,\ i = 1, \ldots, N$
- Initial K is the largest of $K_1 + K_0, K_2 + K_0, K_3 + K_0$ with $K_0 = 5$
- z_i are assumed to be drawn from a **Gaussian mixture**:

$$f(z_i|\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K, \boldsymbol{\eta}) = \sum_{k=1}^{K} \eta_k \, \varphi_k(z_i|\boldsymbol{\theta}_k)$$

- $\boldsymbol{\theta}_k = (\mu_k, \sigma_k^2)$ and η_k are the **component specific parameters** and the **component weight** of component k
- **Sparse solutions** w.r.t. K are obtained by choosing an appropriate prior for the component weights $\boldsymbol{\eta}$
- We use a **symmetric Dirichlet prior** with a concentration parameter e_0:

$$p(\eta_1, \ldots, \eta_K|e_0) = \frac{\Gamma(Ke_0)}{\Gamma(e_0)^K} \prod_{k=1}^{K} \eta_k^{e_0 - 1}$$

Model and priors II

- Smaller values of e_0 give **fewer clusters**
- The prior of the component means is **normal**, the prior of the component variances is **inverse Gaussian**

Clustering

- **Data augmentation:** Introduce latent allocation variables $\boldsymbol{S} = (S_1, \ldots, S_N)$ with values in $\{1, \ldots, K\}$ such that for $i = 1, \ldots, N$

$$f(z_i|\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K, S_i = k) = \varphi(z_i|\mu_k, \sigma_k^2), \quad \Pr(S_i = k|\boldsymbol{\eta}) = \eta_k$$

- Initial values of $\boldsymbol{S}$ from k-means clustering (MATLAB function **kmeans**)
- **Estimation:** Generate a Markov chain from the posterior distribution of $\boldsymbol{S}$ by a **Gibbs sampler**
- **Cluster identification:** Choose the configuration of $\boldsymbol{S}$ with the **largest posterior probability**

Markov Chain Monte Carlo

- A Markov chain is a **non-independent** random sample with the **Markov property**:

$$f(X_{t+1}|X_0 = x_0, \ldots, X_t = x_t) = f(X_{t+1}|X_t = x_t)$$

- Depending on how it is generated, a Markov chain may or may not have a **stationary** or **equlibrium distribution**
- Given a **target** distribution $\pi(x)$, Markov Chain Monte Carlo (MCMC) generates a Markov chain with stationary distribution **equal to** $\pi(x)$
- The target distribution does **not** have to be normalized
- MCMC is therefore an indispensable tool in **Bayesian inference**
- There are two basic ways of generating a Markov chain with a given stationary (target) distribution:
  - **Metropolis–Hastings sampling**
  - **Gibbs sampling**

Metropolis–Hastings sampling

- Metropolis–Hastings sampling of a given target distribution $\pi(x)$ works as follows
- Let x_t be the current value of the chain and x' a value drawn from the **proposal density** $p(x)$ which may depend on x_t
- Compute the **acceptance probability**:

$$\alpha = \min\left(1, \frac{\pi(x')p(x_t)}{\pi(x_t)p(x')}\right)$$

- Draw a uniform random number u in the interval [0,1]
- If $u < \alpha$, set $x_{t+1} = x'$; otherwise set $x_{t+1} = x_t$
- If $p(x)$ does **not** depend on x_t, the sampler is an **independence** sampler
- $p(x|x_t)$ is symmetric around x_t, the sampler is a **random walk** sampler

Gibbs sampling

- Useful for sampling from the **unknown joint distribution** of several variables
- The **conditional distribution** of each x_i given all other variables has to be known
- For each variable in turn, draw a random number from this conditional distribution
- The joint distribution is the stationary distribution of the resulting Markov chain

Burn-in and diagnostics

- As it may take some time to reach the stationary distribution it is frequent practice to discard an initial segment of the chain
- This is called **burn-in**
- It is not always clear when or whether the stationary distribution has been reached
- Visual inspection of the chain can show whether the entire support of the target is explored: good vs bad **mixing**
- The **autocorrelation** of the chain can be used to compute an **effective sample size**

Iterative Maximum-Likelihood

- For the most likely numbers of clusters $K = K_1, K_2, K_3, \ldots, K_3 + 5$:
  1. Choose starting values of mixture parameters $(\mu_1, \ldots, \mu_K, \sigma_1^2, \ldots, \sigma_K^2, \eta_1, \ldots, \eta_K)$
  2. Compute the **association probabilities** p_{ik} and p_k:

$$p_{ik} = \frac{\eta_k \, \varphi(z_i; \mu_k, \sigma_k^2)}{\sum_{j=1}^{K} \eta_l \, \varphi(z_i; \mu_j, \sigma_j^2)}, \quad p_k = \sum_{i=1}^{N} p_{ik}$$

  3. **Estimation** of weights and cluster parameters:

$$\eta_k = \frac{p_k}{N}, \; \mu_k = \frac{\sum_{i=1}^{N} p_{ik} z_i}{p_k}, \; \sigma_k^2 = \frac{\sum_{i=1}^{N} p_{ik} (z_i - \mu_k)^2}{p_k}$$

  4. **Repeat** steps 2 and 3 **until convergence**

- Choose the clustering with the **smallest BIC**
- We have used the MATLAB function **fitgmdist**

Feasibility study

Feasibility study

Bunch crossings

- Superimpose K interactions to a bunch crossing
- K is drawn from a Poisson distribution with mean $\lambda = 100$
- The vertex positions $z_k,\ k = 1, \ldots, K$ are distributed independently according to a normal distribution with mean $\mu = 0\,\text{mm}$ and $\sigma = 65\,\text{mm}$
- We have analyzed 600 bunch crossings with about 60000 interactions and about 2 million tracks

Sections

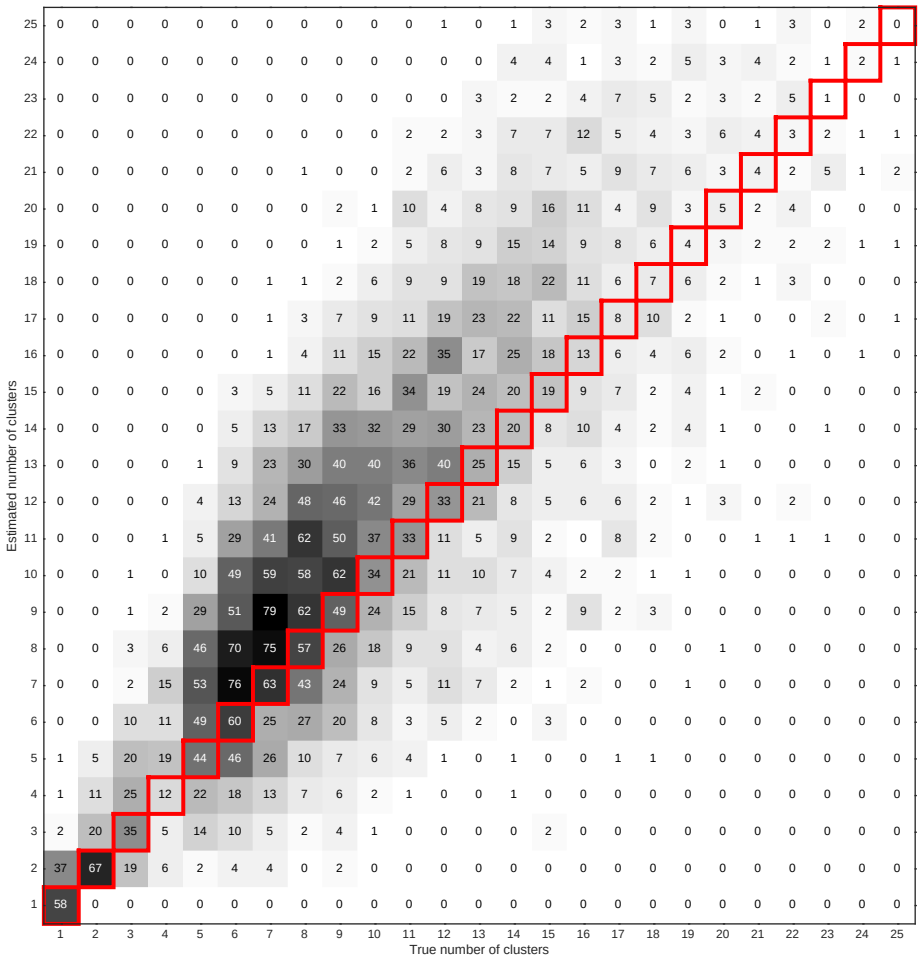
- The z-coordinates of all tracks are filled in a histogram with a bin width of $h = 1\,\text{mm}$
- Boundaries of basic sections are defined by empty bins
- A fixed number (10) of basic sections are combined to the final sections
- Clustering proceeds independently in each final section

Feasibility study

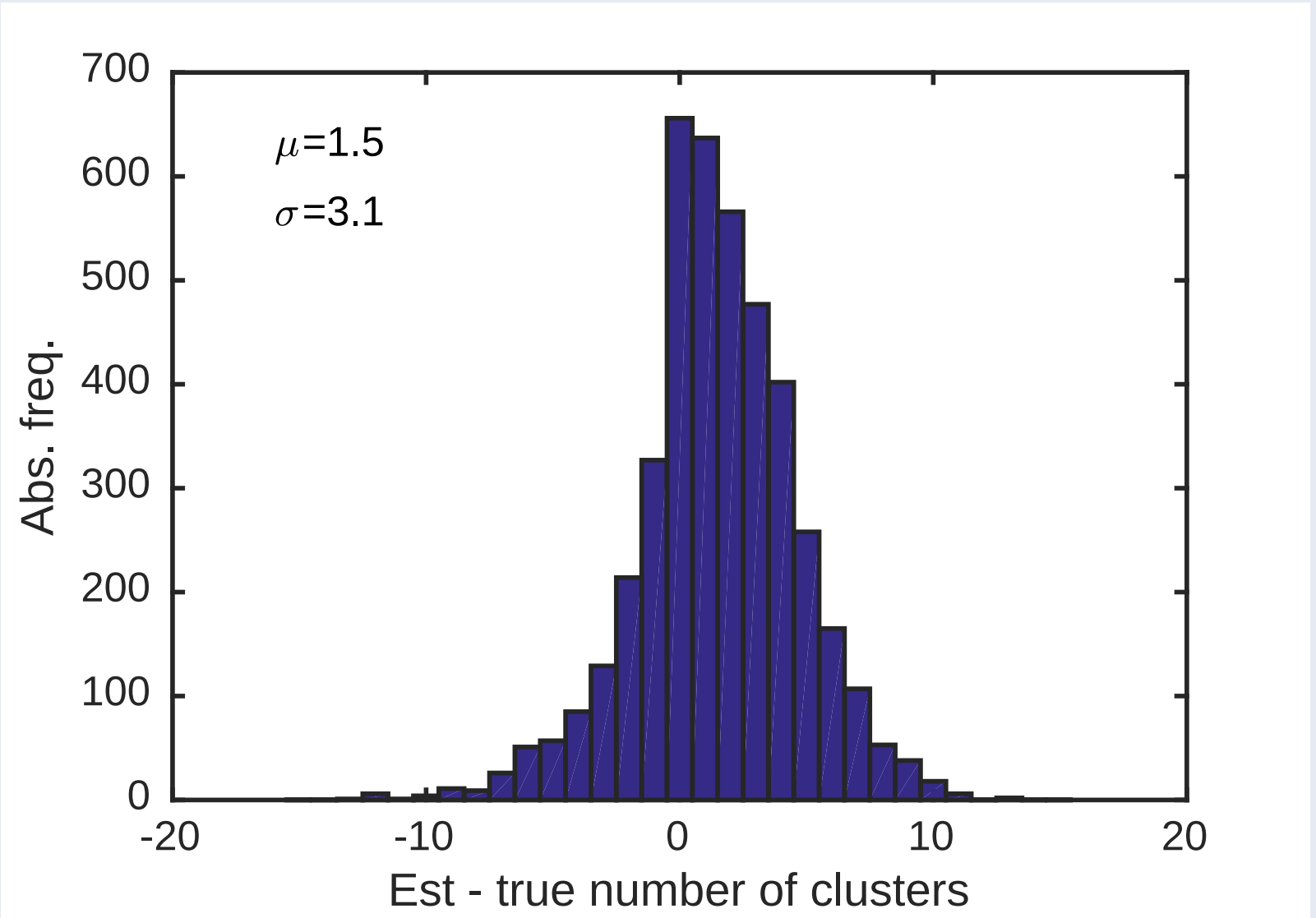
Simulation Runs

- **Run EM:** EM algorithm
- **Run MB1:** Model-based clustering, $e_0 = 0.1$, long MC (1000+5000)
- **Run MB2:** Model-based clustering, $e_0 = 1$, long MC (1000+5000)
- **Run MB3:** Model-based clustering, $e_0 = 1$, short MC (500+1000)

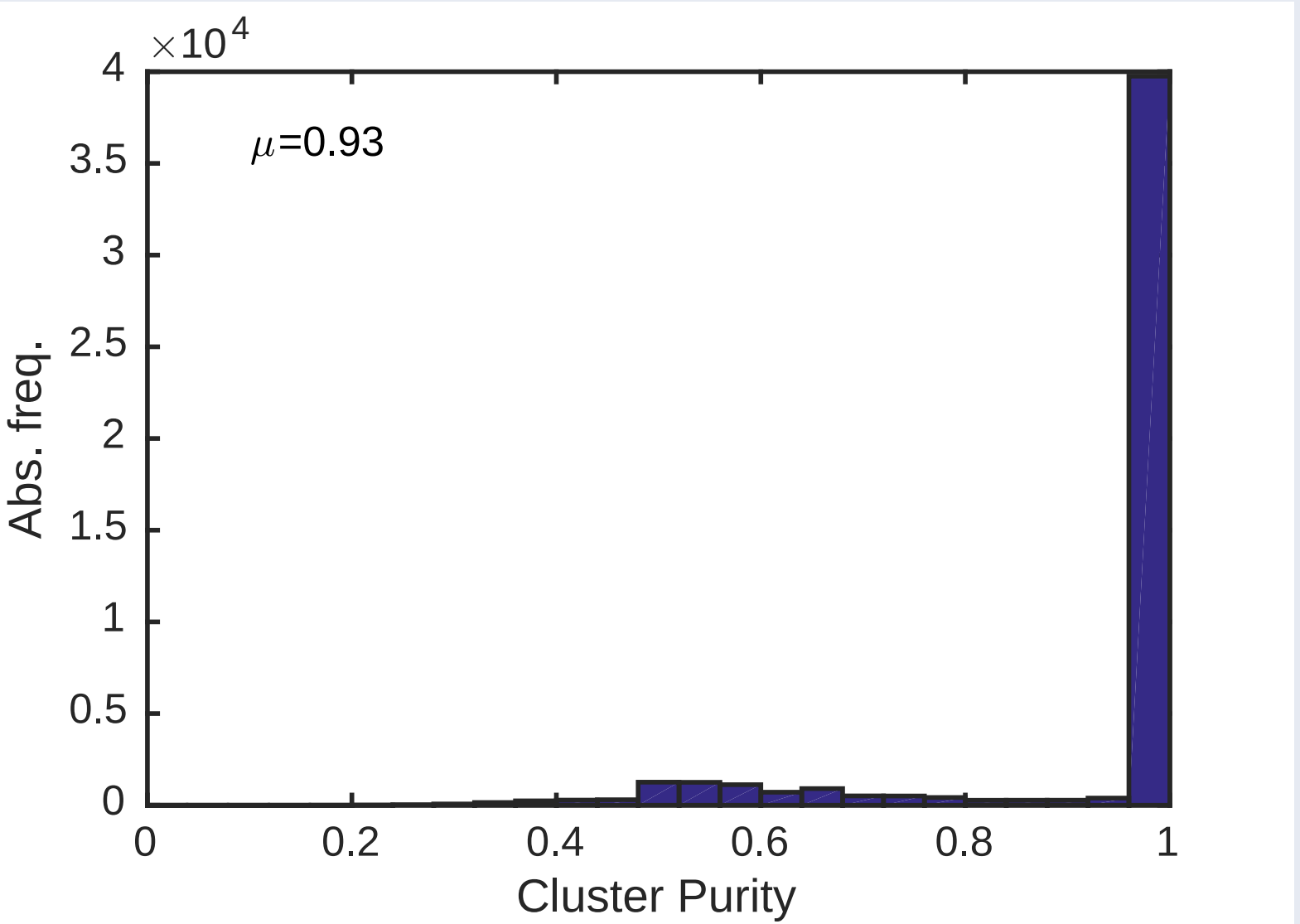
Run EM: true vs estimated cluster number



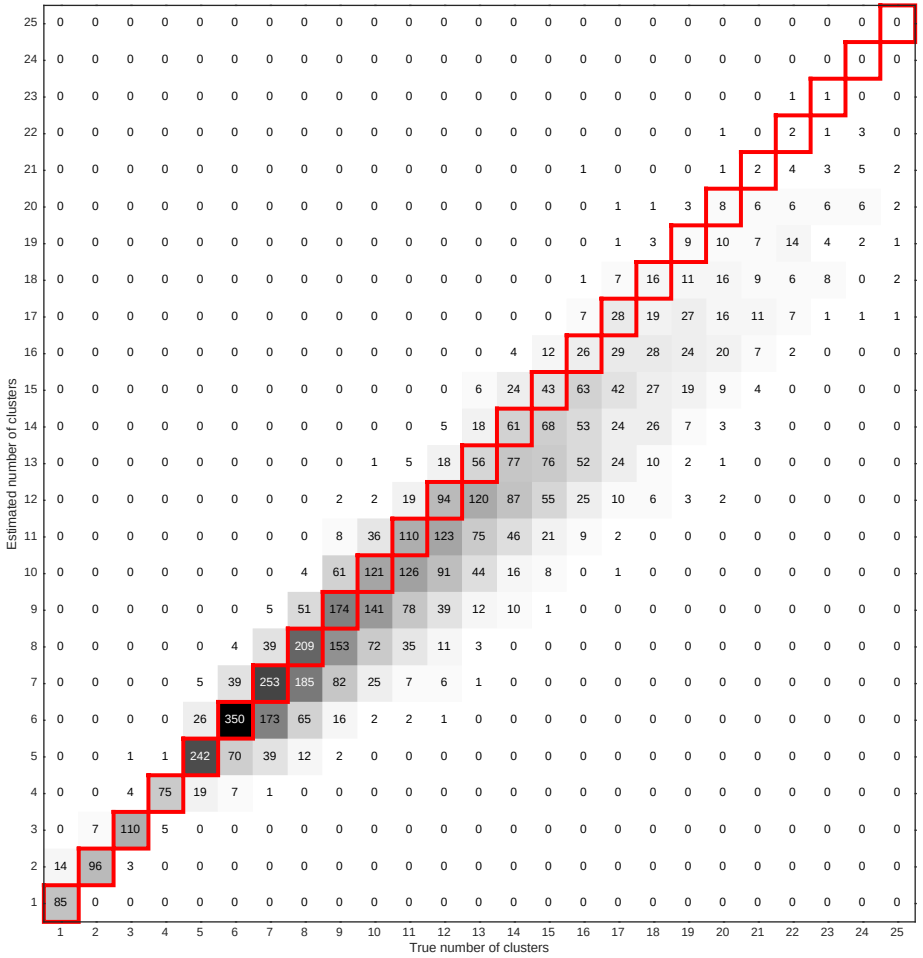
Run EM: estimated minus true cluster number



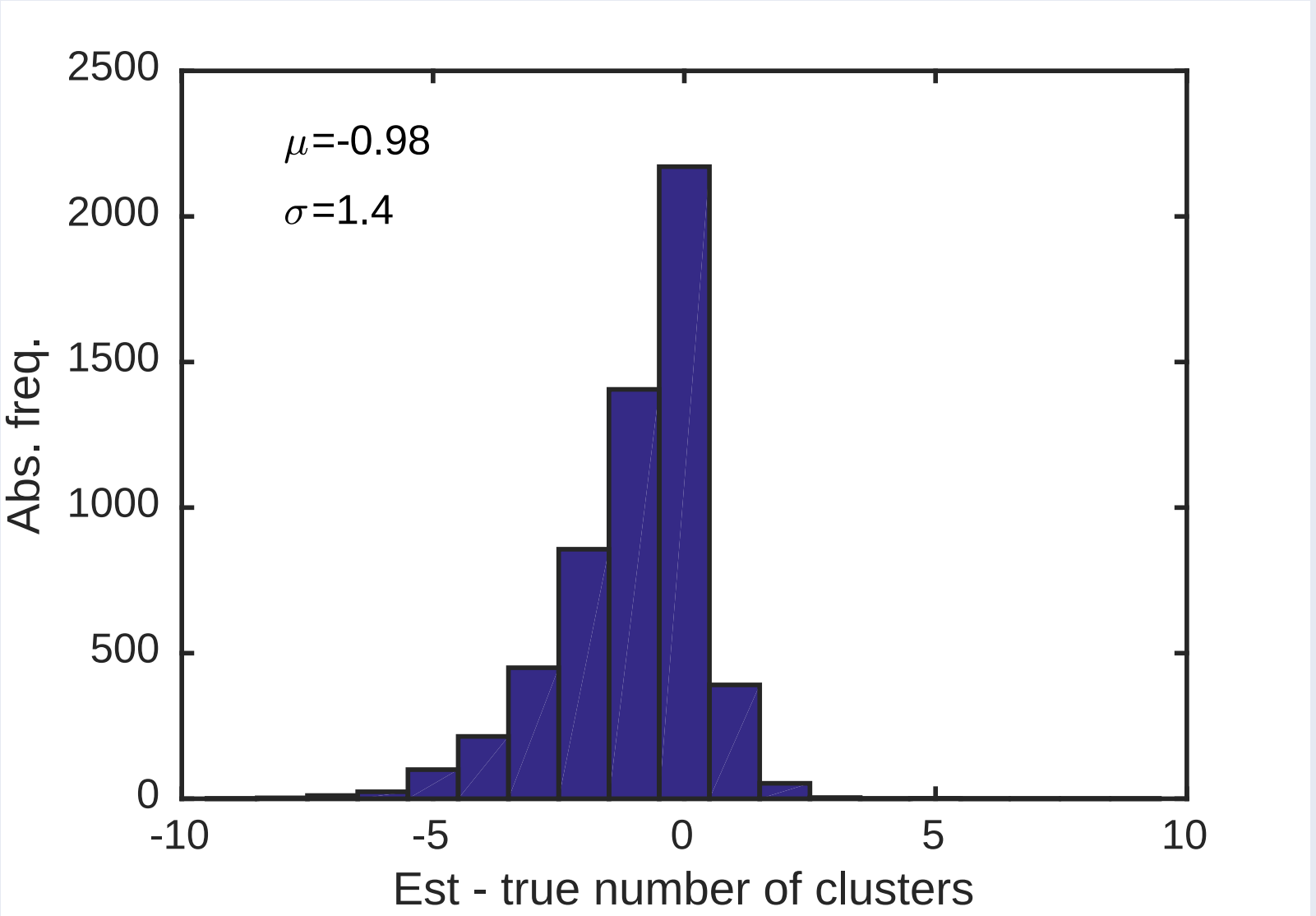
Run EM: cluster purity



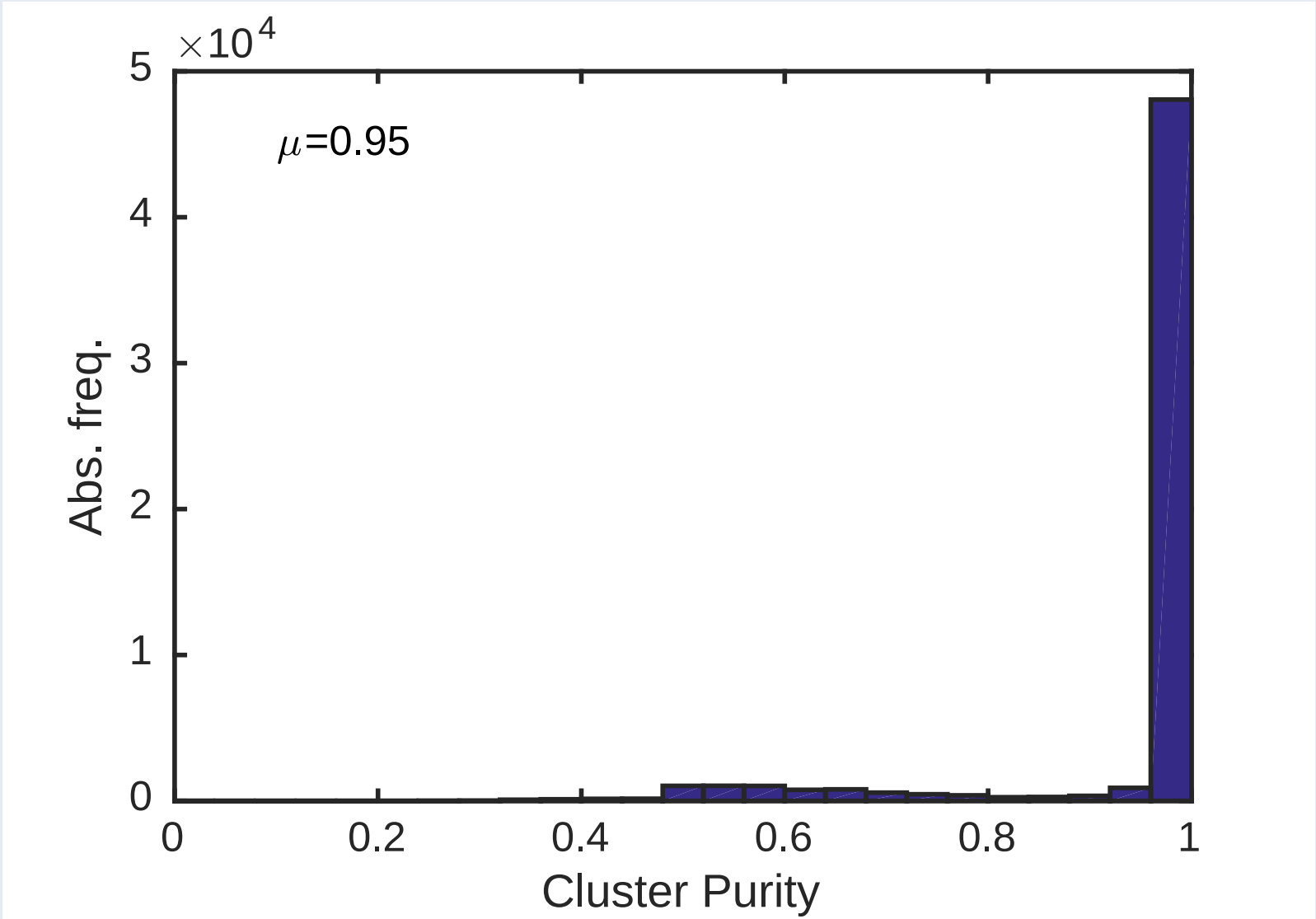
Run MB1: true vs estimated cluster number



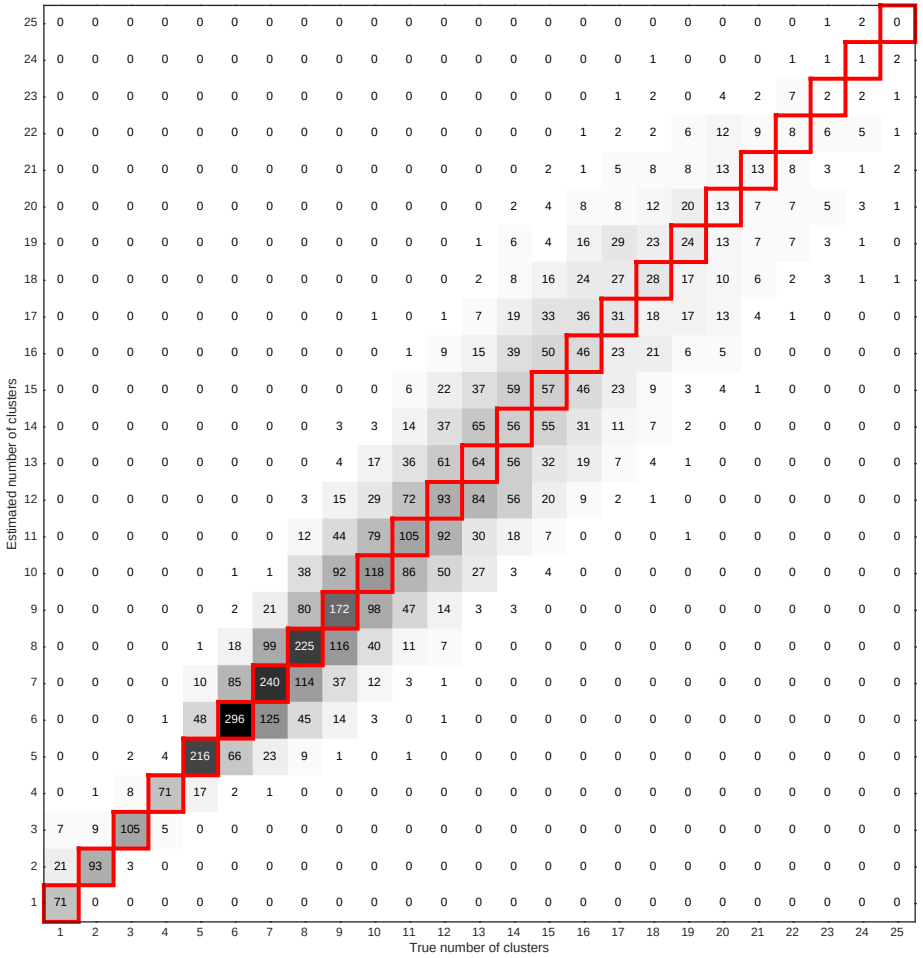
Run MB1: estimated minus true cluster number



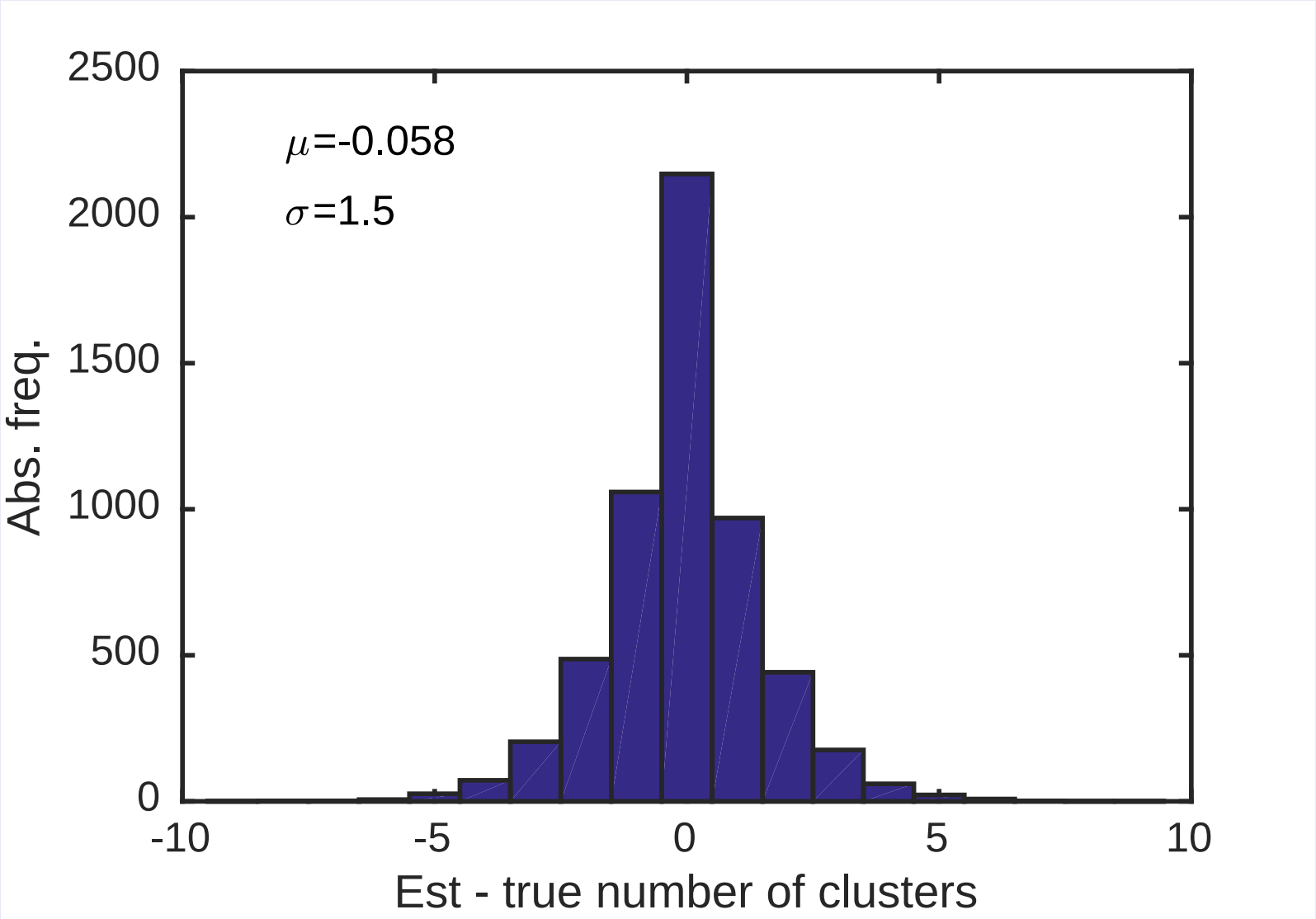
Run MB1: cluster purity



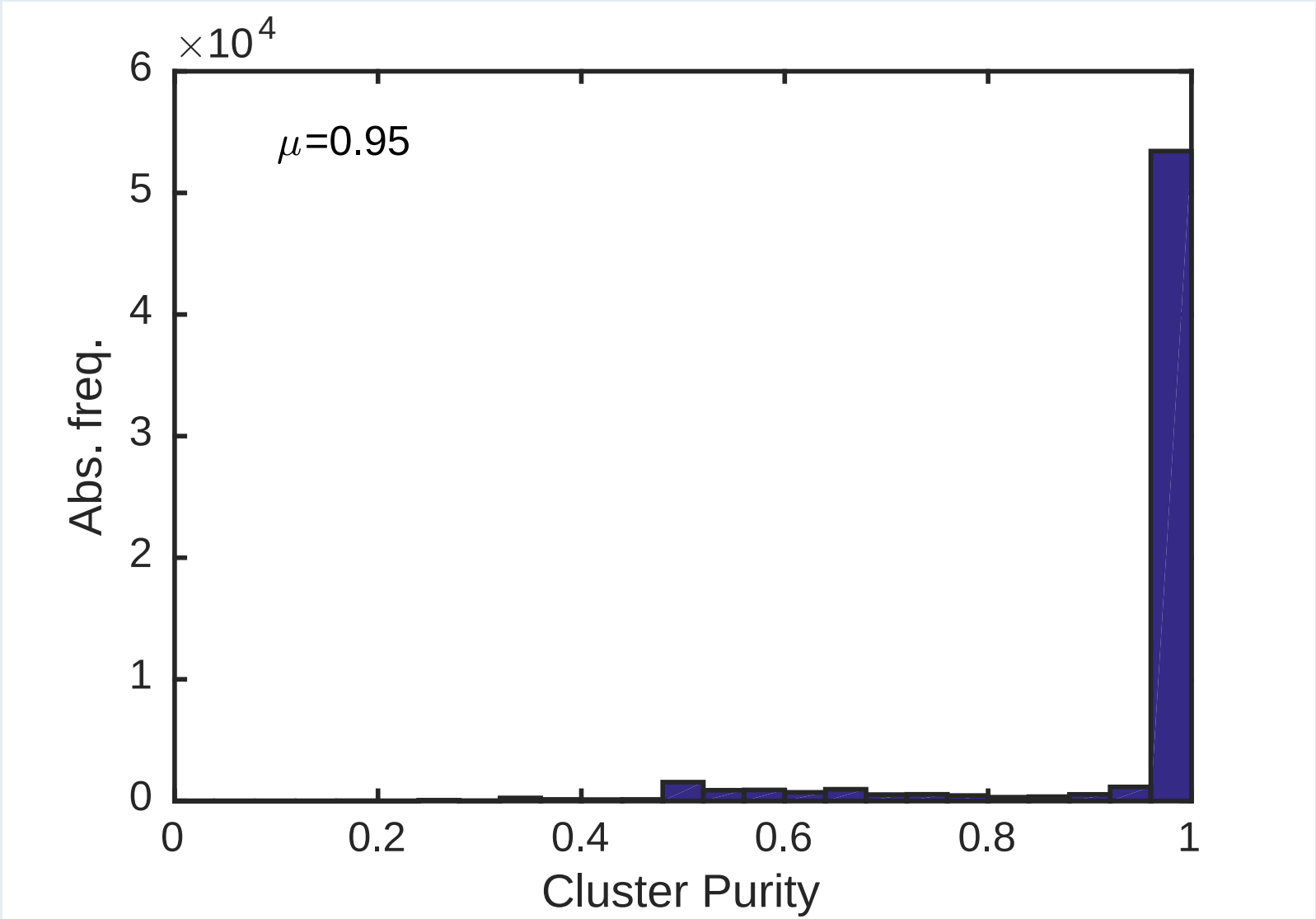
Run MB2: true vs estimated cluster number



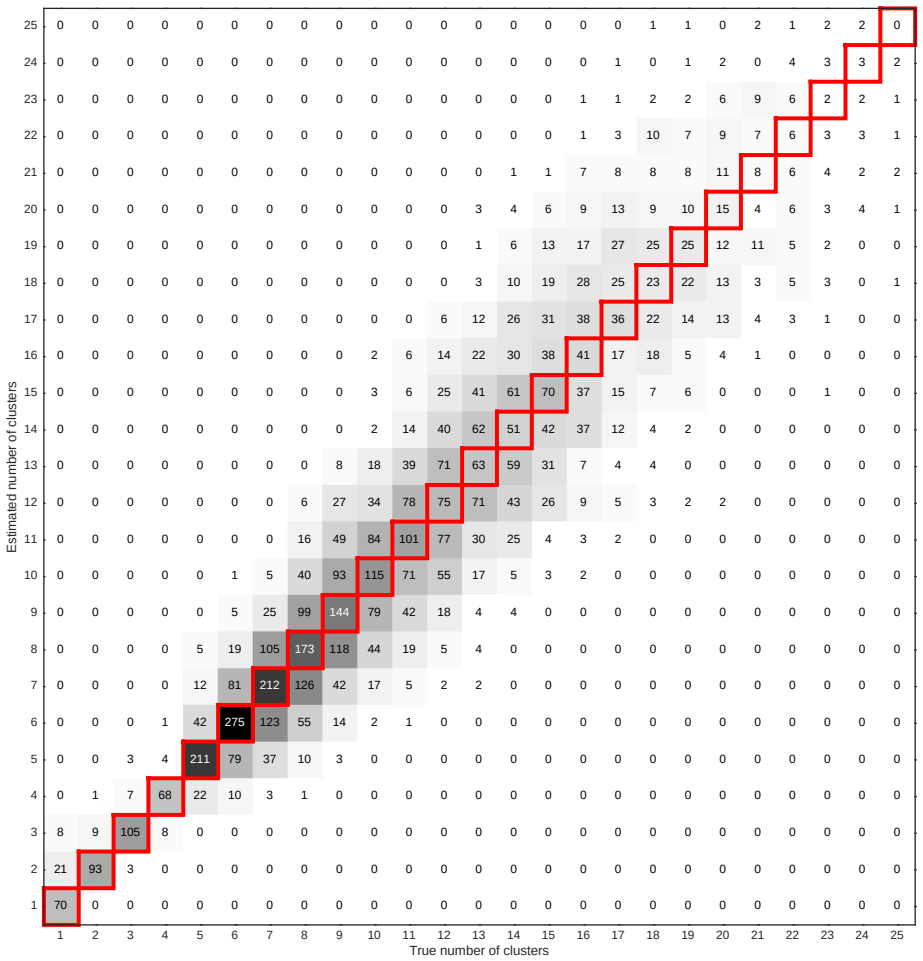
Run MB2: estimated minus true cluster number



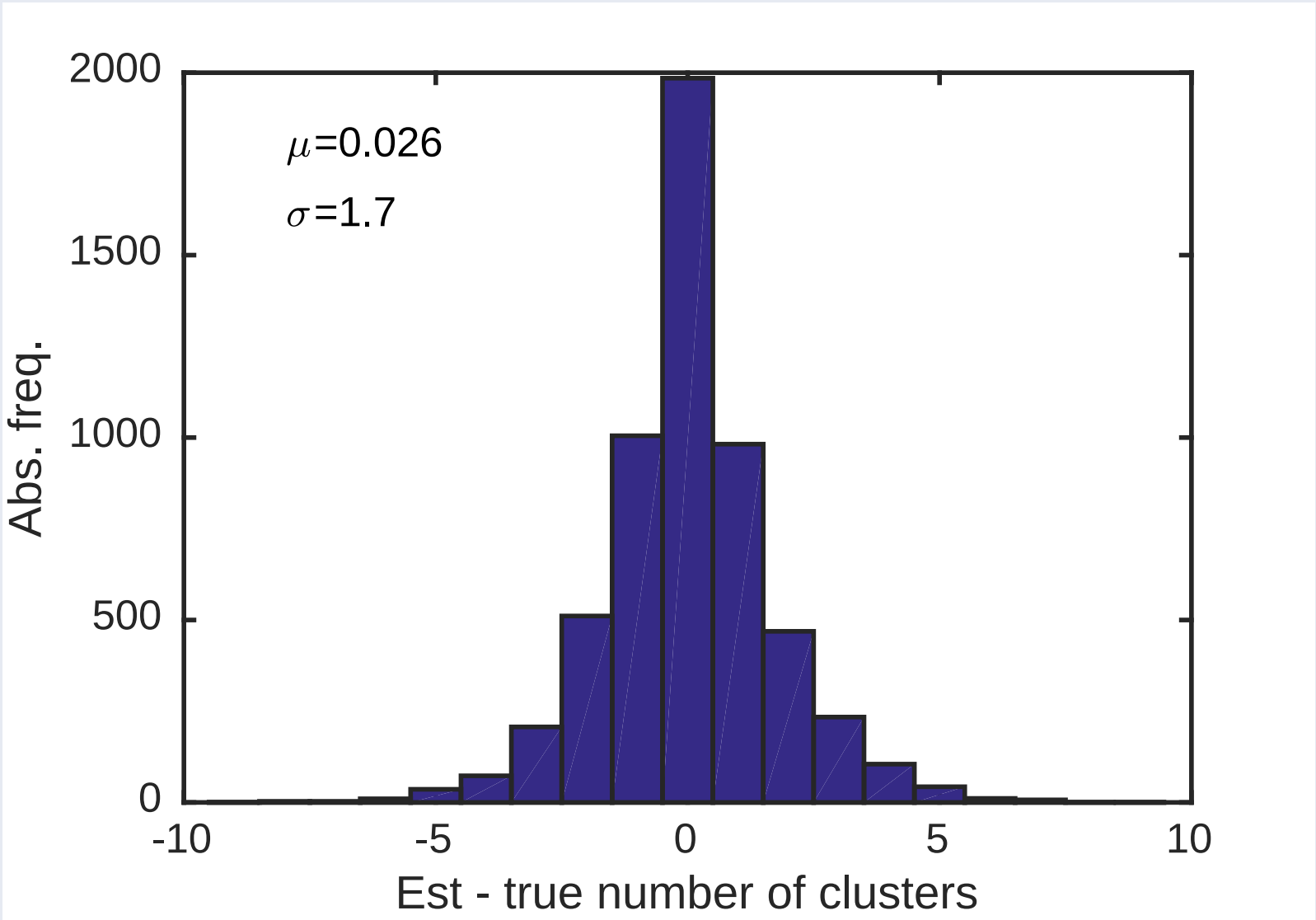
Run MB2: cluster purity



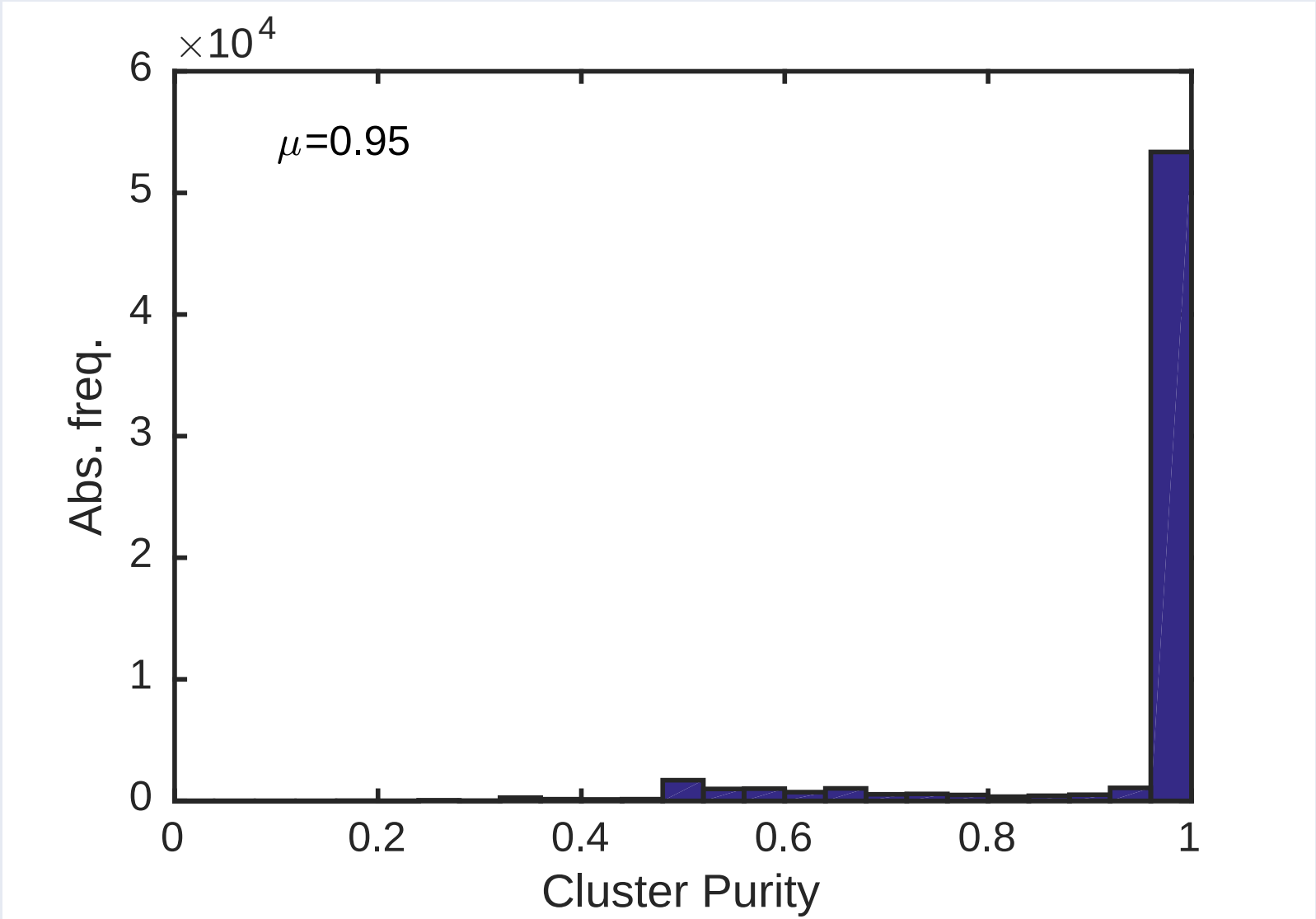
Run MB3: true vs estimated cluster number



Run MB3: estimated minus true cluster number



Run MB3: cluster purity



Discussion and outlook

Discussion and outlook

Pros

- Prior information on number of tracks per vertex and vertex spread is used
- This information can be extracted from runs with low luminosity, where vertex finding is easy
- Only the expected (average) number of vertices depends on the luminosity
- With a bit of fiddling we get the correct average number of vertices, in contrast to EM
- Further tuning of the parameters of the priors such as e_0 possible, e.g. using *spearmint* oder *hyperopt*
- It's fun to try . . . especially if you are bored by all this Kalman filter stuff

Cons

- MCMC sampling is slow by HEP standards:
  clustering takes several seconds rather than a fraction of a second
- Fine-tuning may take a long time
- Hardly suitable for standard vertex finding

Discussion and outlook

Pros

- Prior information on number of tracks per vertex and vertex spread is used
- This information can be extracted from runs with low luminosity, where vertex finding is easy
- Only the expected (average) number of vertices depends on the luminosity
- With a bit of fiddling we get the correct average number of vertices, in contrast to EM
- Further tuning of the parameters of the priors such as e_0 possible, e.g. using *spearmint* oder *hyperopt*
- It's fun to try . . . especially if you are bored by all this Kalman filter stuff

Cons

- MCMC sampling is slow by HEP standards:
  clustering takes several seconds rather than a fraction of a second
- Fine-tuning may take a long time
- Maybe the Kalman filter isn't that bad after all . . .

Possible other applications

- There are interesting problems with **fewer observations** and **fewer clusters**
- **Ring finding in RICH detectors**
  - Model is circle (or ellipse) plus radial uncertainty
  - Put prior distribution on radius
  - Gives a ring-shaped prior
- **Cluster finding in calorimeters**
  - Prior knowledge of the cluster shapes can be injected into the clustering
  - Prior information would have to depend on the type and the location of the shower
- We would have to move to **non-Gaussian models**, possibly more complex samplers
- Merits some further investigation

References

References

Model-based clustering

- Banfield, J.D., Raftery, A.E.: *Model-based Gaussian and non-Gaussian clustering*. Biometrics 49, 803–821 (1993)
- *One of the first papers on the subject*

Asymptotic behaviour

- Rousseau, J., Mengersen, K.: *Asymptotic behaviour of the posterior distribution in overfitted mixture models*.
  J. R. Stat. Soc. B 73(5), 689–710 (2011)
- *Hardcore asymptotic statistics, not for the faint of heart*

Sparse clustering

- Malsiner-Walli, G., Frühwirth-Schnatter, S., Grün, B.:
  Model-based clustering based on sparse finite Gaussian mixtures.
  Statistics and Computing 26(1), 303–324 (2016)
- *Basis of this talk, many more useful references*

References

Everything you always wanted to know about mixture models (but were afraid to ask . . .)

- Frühwirth-Schnatter, S.: *Finite Mixture and Markov Switching Models*, Springer, New York (2006)
- *Expensive but comprehensive*

Advertisement

Poster 1

Constrained fits with non-Gaussian distributions

Rudolf Frühwirth[1], Oliver Cencic[2]

[1] Institute of High Energy Physics, Austrian Academy of Sciences, Vienna, Austria
[2] Institute of Water Quality, Resources and Waste Management, TU Wien, Vienna, Austria

- Shows how to impose linear or non-linear constraints on non-Gaussian data
- Independence sampler is used to draw from the posterior distribution

Poster 2

A new Riemann fit for circular tracks

Rudolf Frühwirth[1], Are Strandlie[2]

[1] Institute of High Energy Physics, Austrian Academy of Sciences, Vienna, Austria
[2] Norwegian University of Science and Technology, Gjøvik, Norway

- Shows how to improve the resolution of the Riemann circle fit following a proposal by Chernov

The End

Thank you for your attention!