Image analysis

The k-nearest-neighbor method

The *k-nearest-neighbor* (knn) procedure has been used in data analysis and machine learning communities as a quick way to classify objects into predefined groups.

This approach requires a training dataset where both the class y and the vector $\mathbf{x}$ of characteristics (or covariates) of each observation are known.

The k-nearest-neighbor method

The training dataset $(y_i, \mathbf{x}_i)_{1 \leq i \leq n}$ is used by the k-nearest-neighbor procedure to predict the value of y_{n+1} given a new vector of covariates $\mathbf{x}_{n+1}$ in a very rudimentary manner.

The predicted value of y_{n+1} is simply the most frequent class found amongst the k nearest neighbors of $\mathbf{x}_{n+1}$ in the set $(\mathbf{x}_i)_{1 \leq i \leq n}$.

The k-nearest-neighbor method

The classical knn procedure does not involve much calibration and requires no statistical modeling at all!

There exists a Bayesian reformulation.

vision dataset (1)

vision: 1373 color pictures, described by 200 variables rather than by the whole table of 500×375 pixels

Four classes of images: class C_1 for motorcycles, class C_2 for bicycles, class C_3 for humans and class C_4 for cars

vision dataset (2)

Typical issue in computer vision problems: build a classifier to identify a picture pertaining to a specific topic without human intervention

We use about half of the images (648 pictures) to construct the training dataset and we save the 689 remaining images to test the performance of our procedures.

vision dataset (3)



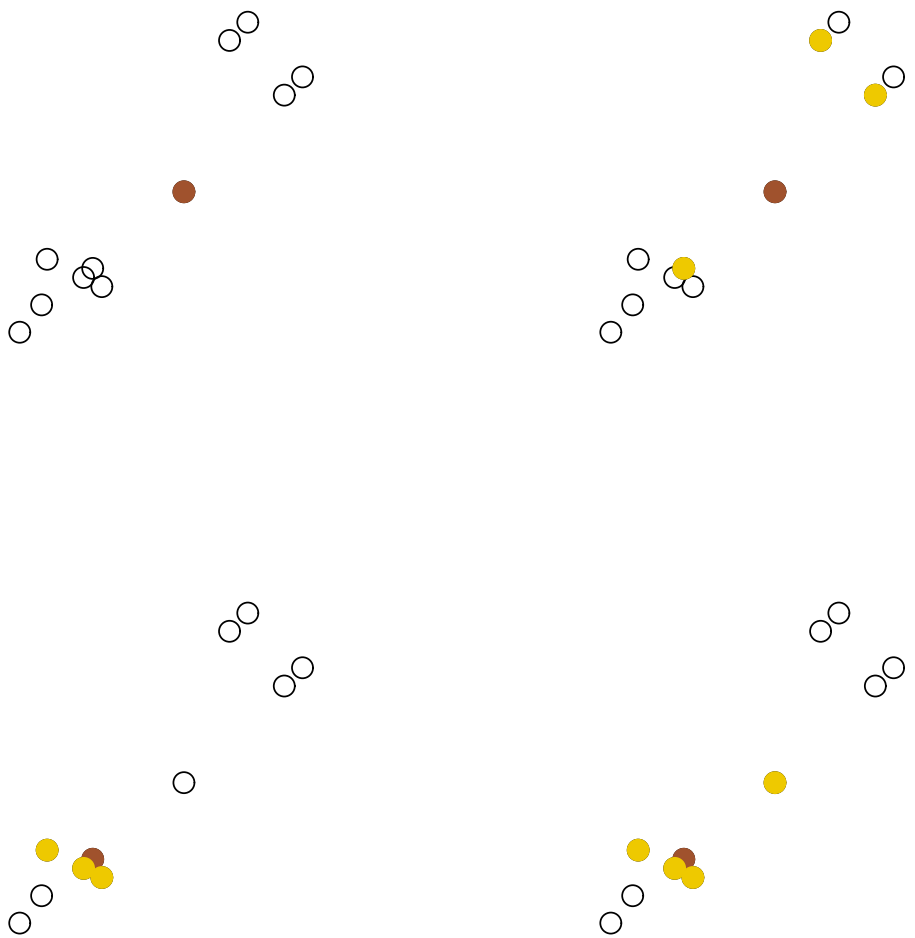

A probabilistic version of the knn methodology (1)

Symmetrization of the neighborhood relation: If $\mathbf{x}_i$ belongs to the k-nearest-neighborhood of $\mathbf{x}_j$ and if $\mathbf{x}_j$ does not belong to the k-nearest-neighborhood of $\mathbf{x}_i$, the point $\mathbf{x}_j$ is added to the set of neighbors of $\mathbf{x}_i$

Notation: $i \sim_k j$

The transformed set of neighbors is then called the *symmetrized k-nearest-neighbor system.*

A probabilistic version of the knn methodology (2)



A probabilistic version of the knn methodology (3)

$$\mathbb{P}(y_i = C_j | \mathbf{y}_{-i}, \mathbf{X}, \beta, k) = \frac{\exp\left(\beta \sum_{\ell \sim_k i} \mathbb{I}_{C_j}(y_\ell) \Big/ N_k\right)}{\sum_{g=1}^{G} \exp\left(\beta \sum_{\ell \sim_k i} \mathbb{I}_{C_g}(y_\ell) \Big/ N_k\right)},$$

N_k being the average number of neighbours over all x_i's, $\beta > 0$ and

$$\mathbf{y}_{-i} = (y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n) \quad \text{and} \quad \mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}.$$

A probabilistic version of the knn methodology (4)

- β grades the influence of the prevalent neighborhood class ;
- the probabilistic knn model is conditional on the covariate matrix $\mathbf{X}$;
- the frequencies $n_1/n, \ldots, n_G/n$ of the classes within the training set are representative of the marginal probabilities $p_1 = \mathbb{P}(y_i = C_1), \ldots, p_G = \mathbb{P}(y_i = C_G)$:

If the marginal probabilities p_g are known and different from n_g/n $\Longrightarrow$ reweighting the various classes according to their true frequencies

Bayesian analysis of the knn probabilistic model

From a Bayesian perspective, given a prior distribution $\pi(\beta, k)$ with support $[0, \beta_{\max}] \times \{1, \ldots, K\}$, the marginal predictive distribution of y_{n+1} is

$$\mathbb{P}(y_{n+1} = C_j | \mathbf{x}_{n+1}, \mathbf{y}, \mathbf{X}) = \sum_{k=1}^{K} \int_0^{\beta_{\max}} \mathbb{P}(y_{n+1} = C_j | \mathbf{x}_{n+1}, \mathbf{y}, \mathbf{X}, \beta, k) \pi(\beta, k | \mathbf{y}, \mathbf{X}) \, \mathrm{d}\beta \,,$$

where $\pi(\beta, k | \mathbf{y}, \mathbf{X})$ is the posterior distribution given the training dataset $(\mathbf{y}, \mathbf{X})$.

Unknown normalizing constant

Major difficulty: it is impossible to compute $f(\mathbf{y}|\mathbf{X},\beta,k)$

Use instead the *pseudo-likelihood* made of the product of the full conditionals

$$\widehat{f}(\mathbf{y}|\mathbf{X},\beta,k) = \prod_{g=1}^{G} \prod_{y_i=C_g} \mathbb{P}(y_i = C_g|\mathbf{y}_{-i},\mathbf{X},\beta,k)$$

Pseudo-likelihood approximation

Pseudo-posterior distribution

$$\widehat{\pi}(\beta, k|\mathbf{y}, \mathbf{X}) \propto \widehat{f}(\mathbf{y}|\mathbf{X}, \beta, k)\pi(\beta, k)$$

Pseudo-predictive distribution

$$\widehat{\mathbb{P}}(y_{n+1} = C_j|\mathbf{x}_{n+1}, \mathbf{y}, \mathbf{X}) =$$
$$\sum_{k=1}^{K} \int_0^{\beta_{\max}} \mathbb{P}(y_{n+1} = C_j|\mathbf{x}_{n+1}, \mathbf{y}, \mathbf{X}, \beta, k)\widehat{\pi}(\beta, k|\mathbf{y}, \mathbf{X})\, \mathrm{d}\beta\,.$$

MCMC implementation (1)

MCMC approximation is required

Random walk Metropolis–Hastings algorithm

Both β and k are updated using random walk proposals:

(i) for β, logistic tranformation

$$\beta^{(t)} = \beta_{\max} \exp(\theta^{(t)}) \big/ (\exp(\theta^{(t)}) + 1)\,,$$

in order to be able to simulate a normal random walk on the θ's, $\tilde{\theta} \sim \mathcal{N}(\theta^{(t)}, \tau^2)$;

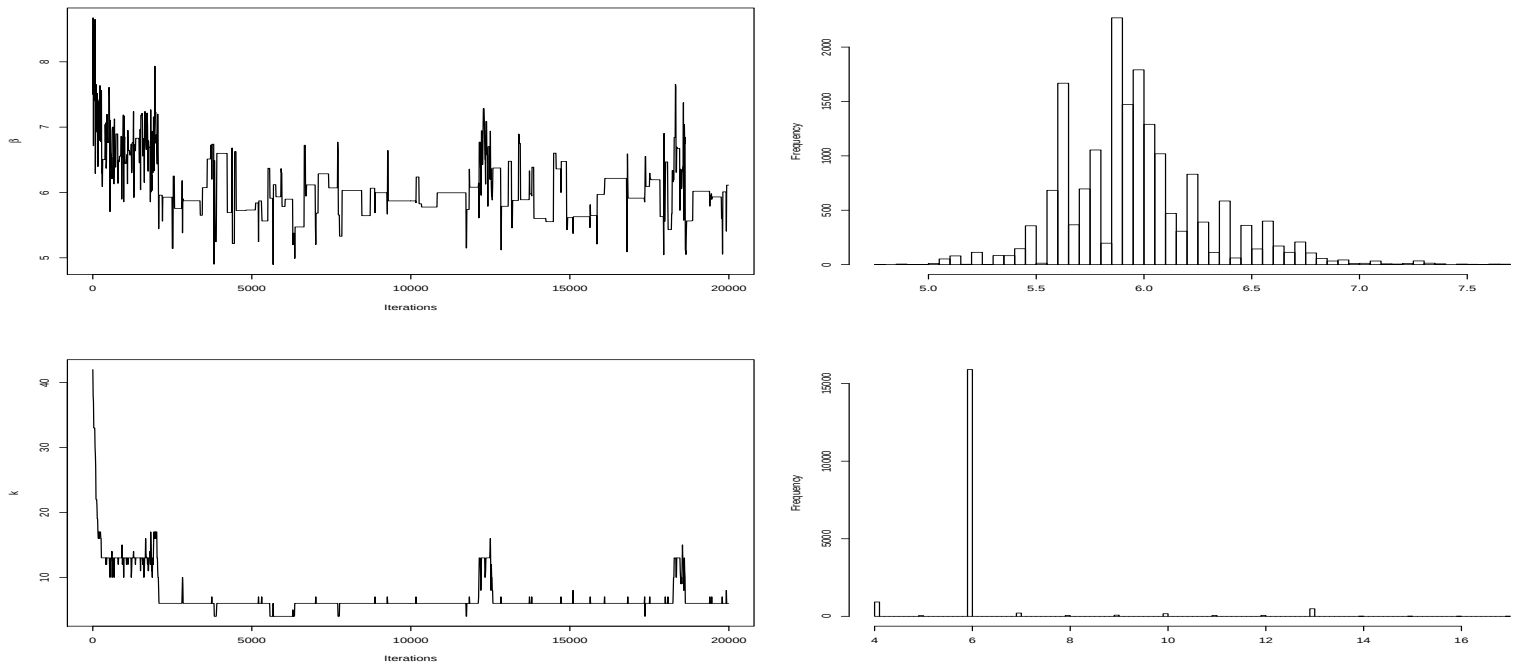
MCMC implementation (2)

(ii) for k, we use a uniform proposal on the r neighbors of $k^{(t)}$, namely on $\{k^{(t)}-r,\ldots,k^{(t)}-1,k^{(t)}+1,\ldots k^{(t)}+r\}\cap\{1,\ldots,K\}$.

Using this algorithm, we can thus derive the most likely class associated with a covariate vector $\mathbf{x}_{n+1}$ from the approximated probabilities,

$$\frac{1}{M}\sum_{i=1}^{M}\widehat{\mathbb{P}}\left(y_{n+1}=l|x_{n+1},y,x,(\beta^{(i)},k^{(i)})\right)\ .$$

MCMC implementation (3)



$\beta_{\max} = 15$, $K = 83$, $\tau^2 = 0.05$ and $r = 1$

k sequences for the knn Metropolis–Hastings β based on $20,000$ iterations

Image Segmentation

This underlying structure of the "true" pixels is denoted by $\mathbf{x}$, while the observed image is denoted by $\mathbf{y}$.

Both objects $\mathbf{x}$ and $\mathbf{y}$ are arrays, with each entry of $\mathbf{x}$ taking a finite number of values and each entry of $\mathbf{y}$ taking real values.

We are interested in the posterior distribution of $\mathbf{x}$ given $\mathbf{y}$ provided by Bayes' theorem, $\pi(\mathbf{x}|\mathbf{y}) \propto f(\mathbf{y}|\mathbf{x})\pi(\mathbf{x})$.

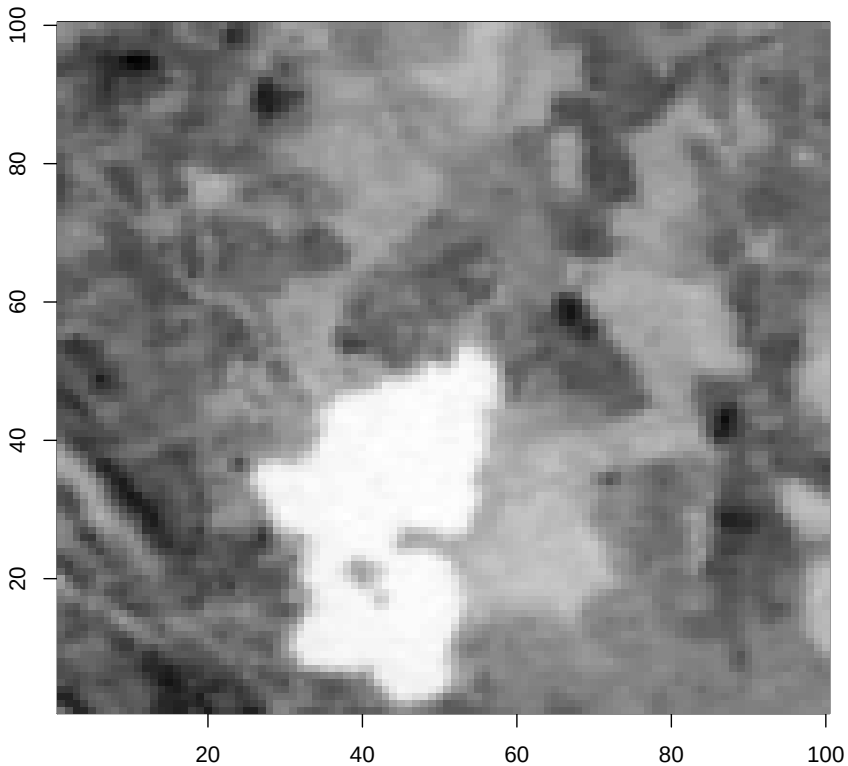
The likelihood, $f(\mathbf{y}|\mathbf{x})$, describes the link between the observed image and the underlying classification.

Menteith dataset (1)

The **Menteith** dataset is a 100×100 pixel satellite image of the lake of Menteith.

The lake of Menteith is located in Scotland, near Stirling, and offers the peculiarity of being called "lake" rather than the traditional Scottish "loch."

Menteith dataset (2)



Satellite image of the lake of Menteith

Random Fields

If we take a lattice $\mathcal{I}$ of sites or pixels in an image, we denote by $i \in \mathcal{I}$ a coordinate in the lattice. The neighborhood relation is then denoted by $\sim$.

A *random field* on $\mathcal{I}$ is a random structure indexed by the lattice $\mathcal{I}$, a collection of random variables $\{x_i; i \in \mathcal{I}\}$ where each x_i takes values in a finite set χ.

Markov Random Fields

Let $\mathbf{x}_{n(i)}$ be the set of values taken by the neighbors of i.

A random field is a *Markov random field* (MRF) if the conditional distribution of any pixel given the other pixels only depends on the values of the neighbors of that pixel; i.e., for $i \in \mathcal{I}$,

$$\pi(x_i|\mathbf{x}_{-i}) = \pi(x_i|\mathbf{x}_{n(i)})\,.$$

Ising Models

If pixels of the underlying (true) image $\mathbf{x}$ can only take two colors (black and white, say), $\mathbf{x}$ is then binary, while $\mathbf{y}$ is a grey-level image.

We typically refer to each pixel x_i as being *foreground* if $x_i = 1$ (black) and *background* if $x_i = 0$ (white).

We have

$$\pi(x_i = j|\mathbf{x}_{-i}) \propto \exp(\beta n_{i,j}) \,, \qquad \beta > 0 \,,$$

where $n_{i,j} = \sum_{\ell \in n(i)} \mathbb{I}_{x_\ell = j}$ is the number of neighbors of x_i with color j.

Ising Models

The *Ising model* is defined via full conditionals

$$\pi(x_i = 1|\mathbf{x}_{-i}) = \frac{\exp(\beta n_{i,1})}{\exp(\beta n_{i,0}) + \exp(\beta n_{i,1})}\,,$$

and the joint distribution satisfies

$$\pi(\mathbf{x}) \propto \exp\left(\beta \sum_{j\sim i} \mathbb{I}_{x_j = x_i}\right)\,,$$

where the summation is taken over all pairs (i, j) of neighbors.

Simulating from the Ising Models

The normalizing constant of the Ising Model is intractable except for very small lattices $\mathcal{I}$...

Direct simulation of $\mathbf{x}$ is not possible!

Ising Gibbs Sampler

Algorithm (**Ising Gibbs Sampler**)

Initialization: For $i \in \mathcal{I}$, generate independently

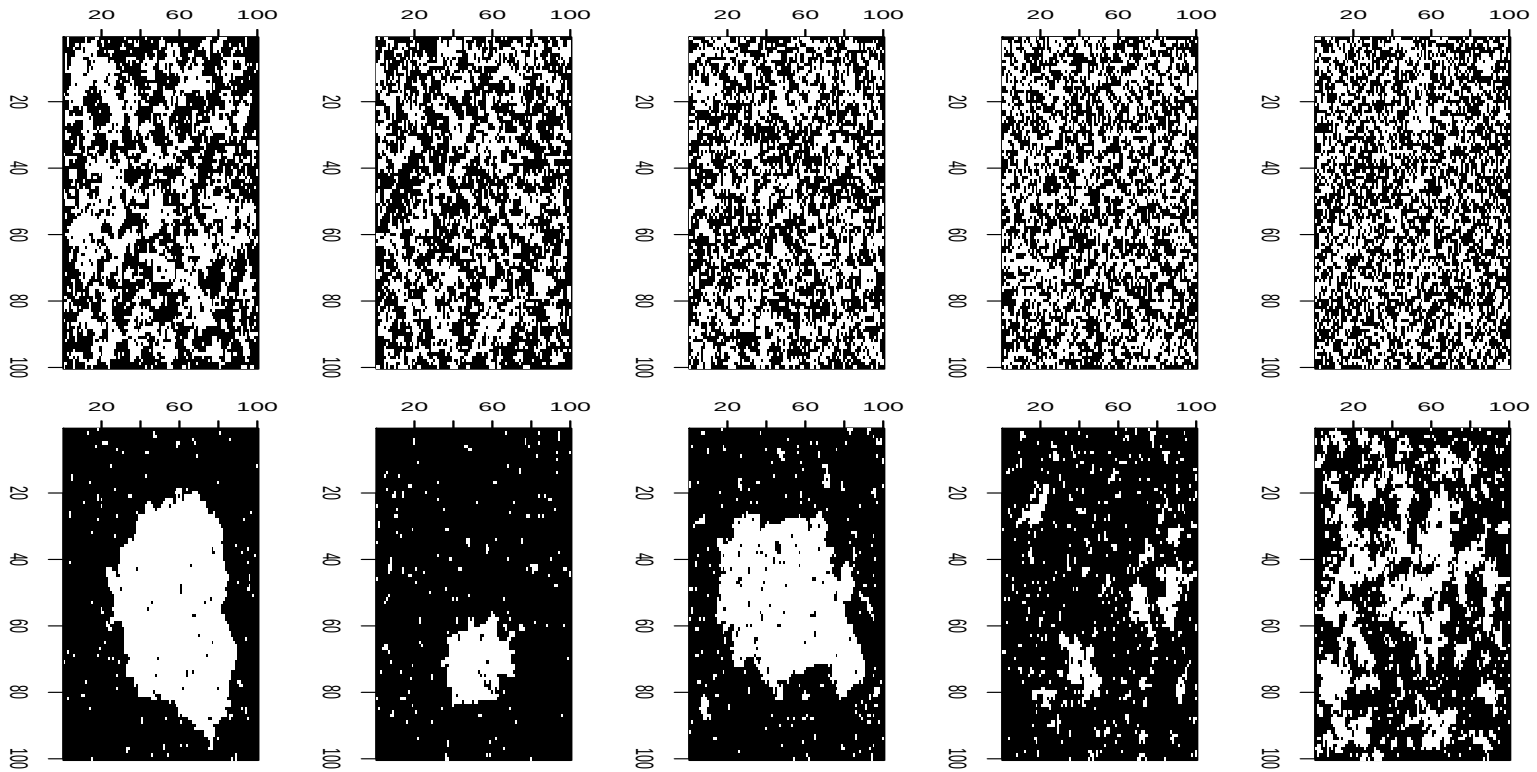
$$x_i^{(0)} \sim \mathscr{B}(1/2)\,.$$

Iteration t $(t \geq 1)$:

1. Generate $\mathbf{u} = (u_i)_{i\in\mathcal{I}}$, a random ordering of the elements of $\mathcal{I}$.
2. For $1 \leq \ell \leq |\mathcal{I}|$, update $n_{u_\ell,0}^{(t)}$ and $n_{u_\ell,1}^{(t)}$, and generate

$$x_{u_\ell}^{(t)} \sim \mathscr{B}\left\{\frac{\exp(\beta n_{u_\ell,1}^{(t)})}{\exp(\beta n_{u_\ell,0}^{(t)}) + \exp(\beta n_{u_\ell,1}^{(t)})}\right\}\,.$$

Ising Gibbs Sampler (2)



Simulations from the Ising model with a four-neighbor neighborhood structure on a 100×100 array after $1,000$ iterations of the Gibbs sampler: β varies in steps of 0.1 from 0.3 to 1.2.

Potts Models

If there are G colors and if $n_{i,g}$ denotes the number of neighbors of $i \in \mathcal{I}$ with color g $(1 \leq g \leq G)$ (that is, $n_{i,g} = \sum_{j \sim i} \mathbb{I}_{x_j = g}$)
In that case, the full conditional distribution of x_i is chosen to satisfy $\pi(x_i = g | \mathbf{x}_{-i}) \propto \exp(\beta n_{i,g})$.
This choice corresponds to the *Potts model*, whose joint density is given by

$$\pi(\mathbf{x}) \propto \exp\left(\beta \sum_{j \sim i} \mathbb{I}_{x_j = x_i} \right) .$$

Simulating from the Potts Models (1)

Algorithm (Potts Metropolis–Hastings Sampler)

Initialization: For $i \in \mathcal{I}$, generate independently

$$x_i^{(0)} \sim \mathcal{U}(\{1, \ldots, G\}) .$$

Iteration t $(t \geq 1)$:

1. Generate $\mathbf{u} = (u_i)_{i \in \mathcal{I}}$ a random ordering of the elements of $\mathcal{I}$.
2. For $1 \leq \ell \leq |\mathcal{I}|$,

   generate

$$\tilde{x}_{u_\ell}^{(t)} \sim \mathcal{U}(\{1, x_{u_\ell}^{(t-1)} - 1, x_{u_\ell}^{(t-1)} + 1, \ldots, G\}) ,$$

Simulating from the Potts Models (2)

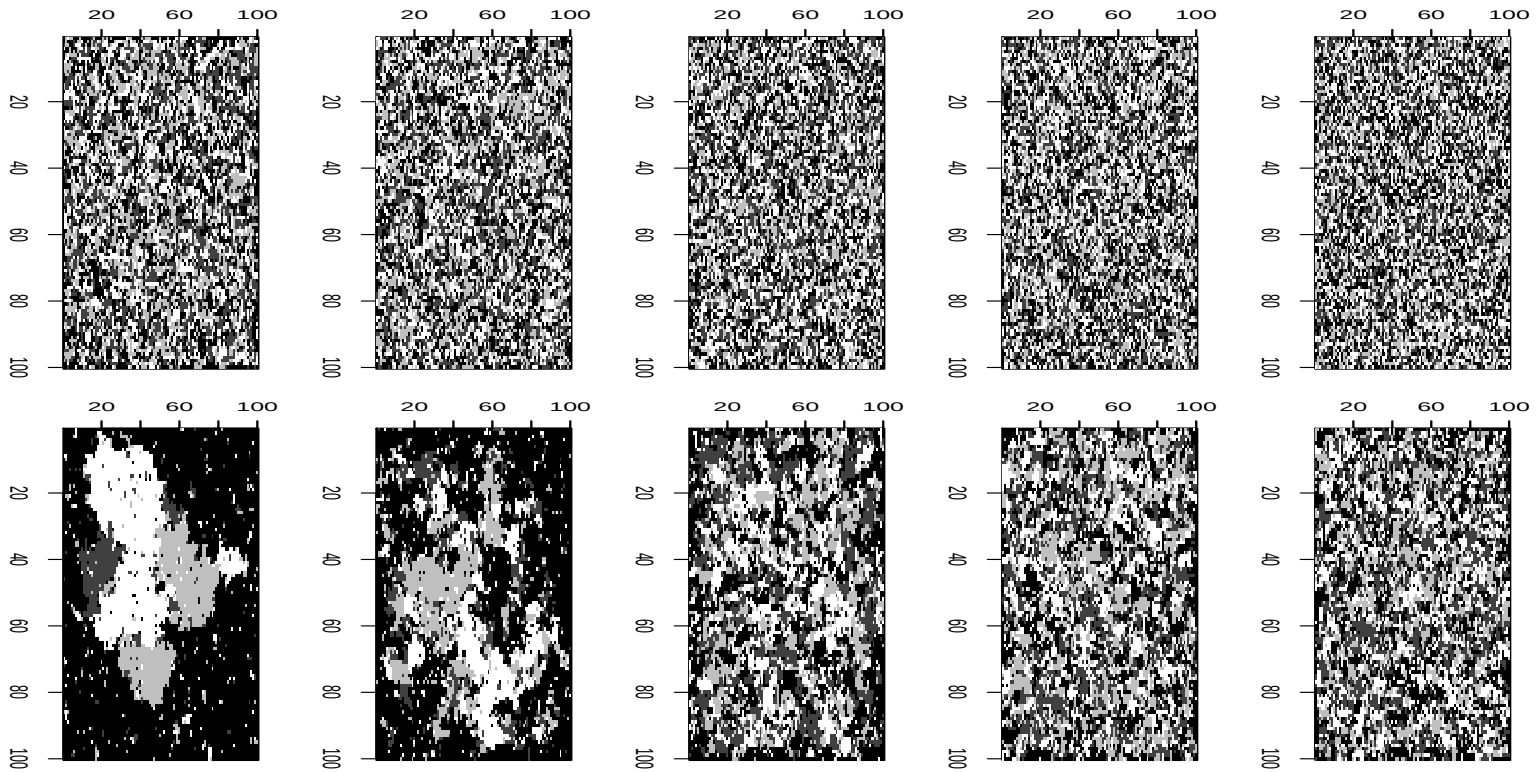
Algorithm (continued)

compute the $n^{(t)}_{u_l,g}$ and

$$\rho_l = \left\{ \exp(\beta n^{(t)}_{u_\ell,\tilde{x}}) / \exp(\beta n^{(t)}_{u_\ell,x_{u_\ell}}) \right\} \wedge 1 \,,$$

and set $x^{(t)}_{u_\ell}$ equal to $\tilde{x}_{u_\ell}$ with probability ρ_l.

Simulating from the Potts Models (3)



Simulations from the Potts model with four grey levels and a four-neighbor neighborhood structure based on 1000 iterations of the Metropolis–Hastings sampler. The parameter β varies in steps of 0.1 from 0.3 to 1.2.

Posterior Inference (1)

The prior on $\mathbf{x}$ is a Potts model with G categories,

$$\pi(\mathbf{x}|\beta) = \frac{1}{Z(\beta)} \exp\left(\beta \sum_{i\in\mathcal{I}} \sum_{j\sim i} \mathbb{I}_{x_j=x_i}\right) ,$$

where $Z(\beta)$ is the normalizing constant.

Given $\mathbf{x}$, we assume that the observations in $\mathbf{y}$ are independent normal random variables,

$$f(\mathbf{y}|\mathbf{x}, \sigma^2, \mu_1, \ldots, \mu_G) = \prod_{i\in\mathcal{I}} \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(y_i - \mu_{x_i})^2\right\} .$$

Posterior Inference (2)

Priors

$$\begin{aligned} \beta &\sim \mathscr{U}([0,2])\,, \\ \boldsymbol{\mu} = (\mu_1, \ldots, \mu_G) &\sim \mathscr{U}(\{\boldsymbol{\mu}\,;\, 0 \leq \mu_1 \leq \ldots \leq \mu_G \leq 255\})\,, \\ \pi(\sigma^2) &\propto \sigma^{-2}\mathbb{I}_{]0,\infty[}(\sigma^2)\,, \end{aligned}$$

the last prior corresponding to a uniform prior on $\log\sigma$.

Posterior Inference (3)

Posterior distribution

$$\pi(\mathbf{x}, \beta, \sigma^2, \boldsymbol{\mu}|\mathbf{y}) \propto \pi(\beta, \sigma^2, \boldsymbol{\mu}) \times \frac{1}{Z(\beta)} \exp\left(\beta \sum_{i\in\mathcal{I}} \sum_{j\sim i} \mathbb{I}_{x_j=x_i}\right) \times \prod_{i\in\mathcal{I}} \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{\frac{-1}{2\sigma^2}(y_i - \mu_{x_i})^2\right\} .$$

Full conditionals (1)

$$\mathbb{P}(x_i = g|\mathbf{y}, \beta, \sigma^2, \boldsymbol{\mu}) \propto \exp\left\{\beta \sum_{j\sim i} \mathbb{I}_{x_j=g} - \frac{1}{2\sigma^2}(y_i - \mu_g)^2\right\},$$

can be simulated directly.

$$n_g = \sum_{i\in\mathcal{I}} \mathbb{I}_{x_i=g} \quad \text{and} \quad s_g = \sum_{i\in\mathcal{I}} \mathbb{I}_{x_i=g} y_i$$

the full conditional distribution of μ_g is a truncated normal distribution on $[\mu_{g-1}, \mu_{g+1}]$ with mean s_g/n_g and variance σ^2/n_g

Full conditionals (2)

The full conditional distribution of σ^2 is an inverse gamma distribution with parameters $|\mathcal{I}|^2/2$ and $\sum_{i\in\mathcal{I}}(y_i - \mu_{x_i})^2/2$.

The full conditional distribution of β is such that

$$\pi(\beta|\mathbf{y}) \propto \frac{1}{Z(\beta)} \exp\left(\beta \sum_{i\in\mathcal{I}} \sum_{j\sim i} \mathbb{I}_{x_j=x_i}\right) .$$

Path sampling Approximation (1)

$$Z(\beta) = \sum_{\mathbf{x}} \exp\{\beta S(\mathbf{x})\} \,,$$

where $S(\mathbf{x}) = \sum_{i\in\mathcal{I}} \sum_{j\sim i} \mathbb{I}_{x_j=x_i}$

$$\begin{aligned} \frac{\mathrm{d}Z(\beta)}{\mathrm{d}\beta} &= Z(\beta) \sum_{\mathbf{x}} S(\mathbf{x}) \frac{\exp(\beta S(\mathbf{x}))}{Z(\beta)} \\ &= Z(\beta)\, \mathbb{E}_\beta[S(\mathbf{X})] \,, \end{aligned}$$

$$\frac{\mathrm{d}\log Z(\beta)}{\mathrm{d}\beta} = \mathbb{E}_\beta[S(\mathbf{X})] \,.$$

Path sampling Approximation (2)

Path sampling identity

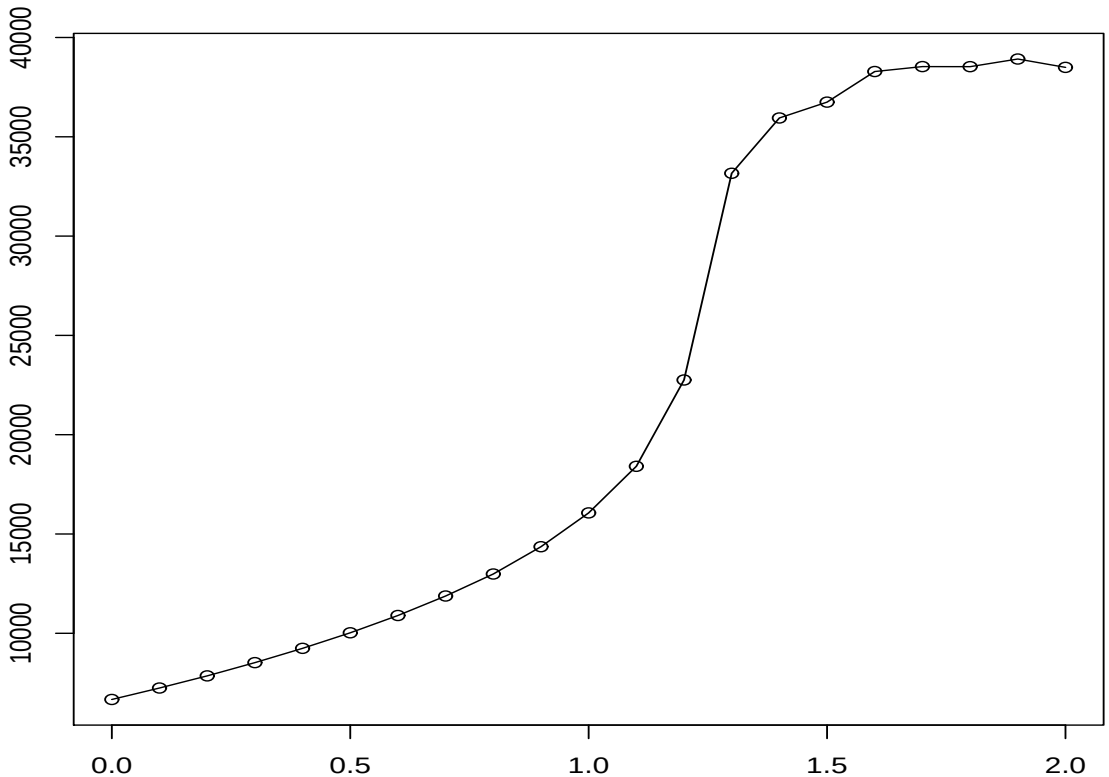
$$\log\{Z(\beta_1)/Z(\beta_0)\} = \int_{\beta_0}^{\beta_1} \mathbb{E}_\beta[S(\mathbf{x})]\mathrm{d}\beta\,.$$

Path sampling Approximation (3)

For a given value of β, $\mathbb{E}_\beta[S(\mathbf{X})]$ can be approximated from an MCMC sequence.

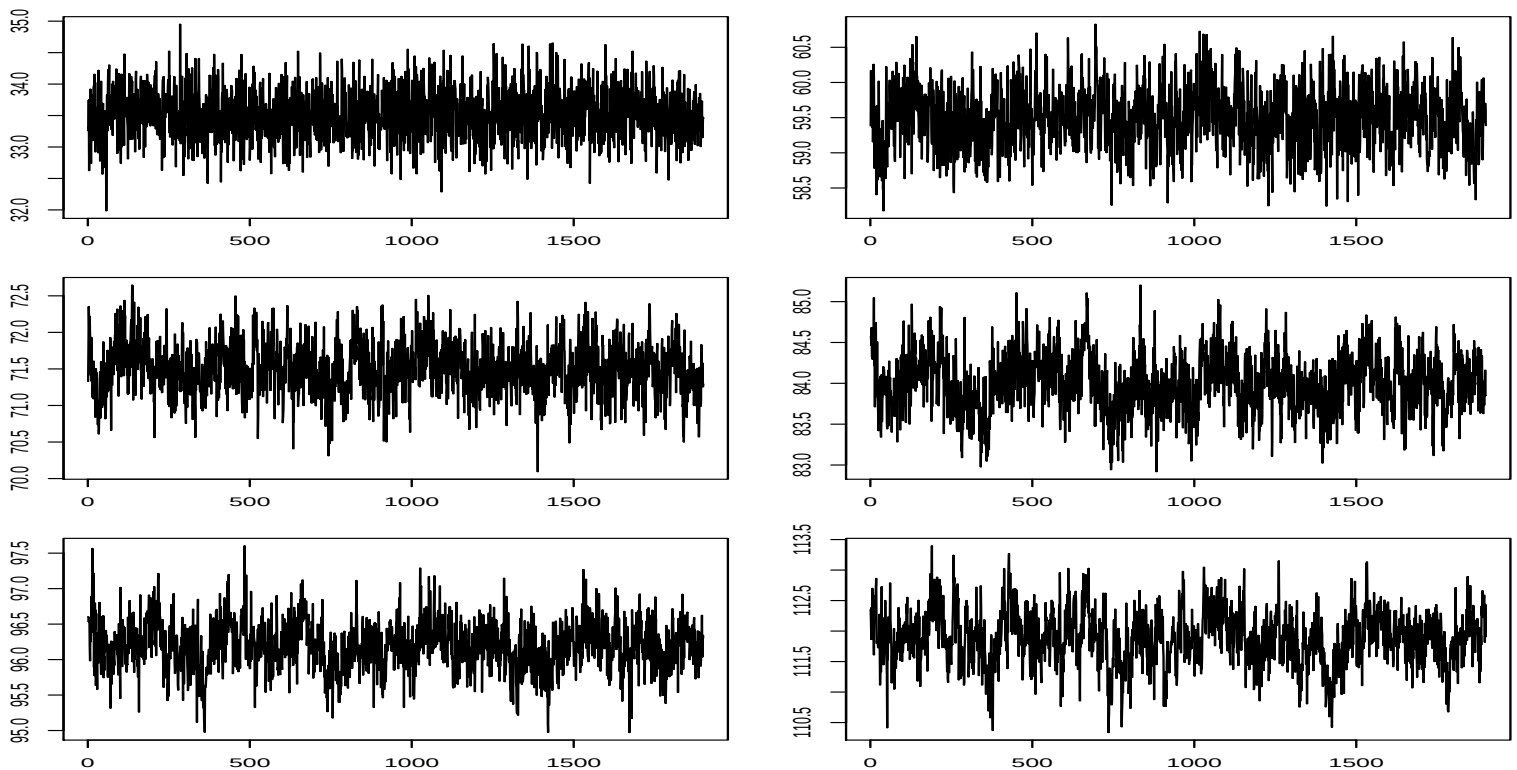
The integral itself can be approximated by computing the value of $f(\beta) = \mathbb{E}_\beta[S(\mathbf{X})]$ for a finite number of values of β and approximating $f(\beta)$ by a piecewise-linear function $\hat{f}(\beta)$ for the intermediate values of β.

Path sampling Approximation (3)



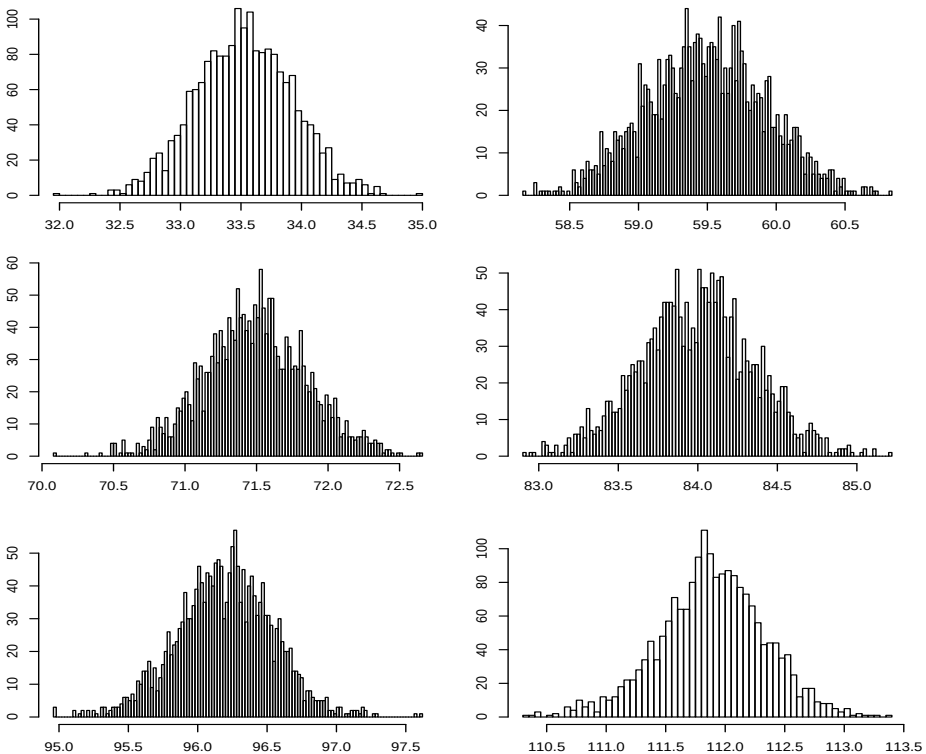
Approximation of $f(\beta)$ for the Potts model on a 100×100 image, a four-neighbor neighborhood, and $G = 6$, based on 1500 MCMC iterations after burn-in.

Posterior Approximation (1)



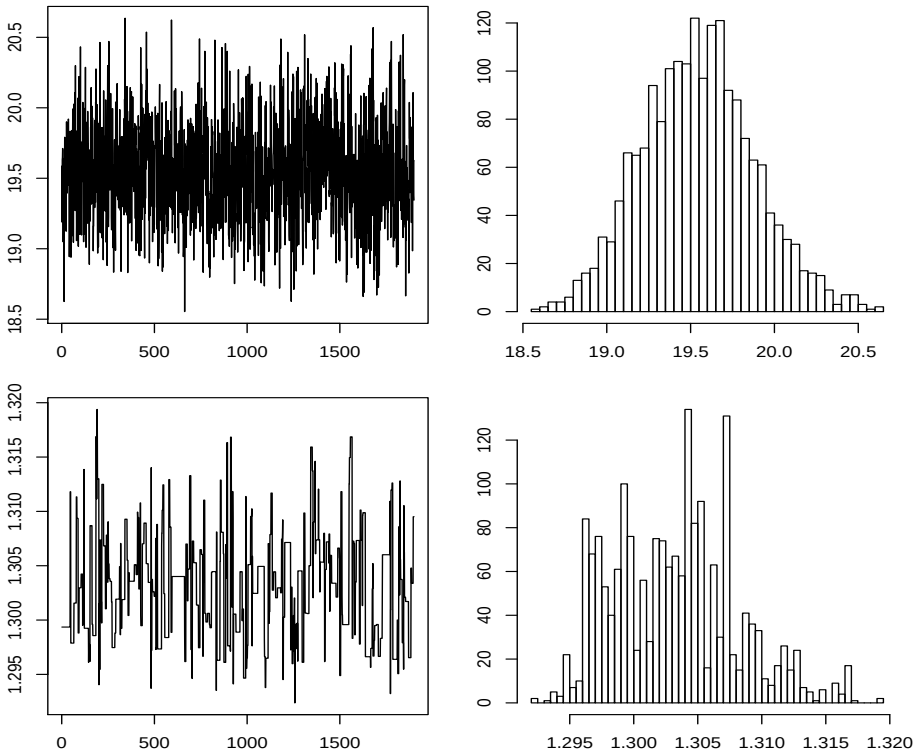
Dataset Menteith: Sequence of μ_g's based on 2000 iterations of the hybrid Gibbs sampler (*read row-wise from* μ_1 *to* μ_6).

Posterior Approximation (2)



Histograms of the μ_g's

Posterior Approximation (3)



Raw plots and histograms of the σ^2's and β's based on 2000 iterations of the hybrid Gibbs sampler

Image segmentation (1)

Based on $(\mathbf{x}^{(t)})_{1\le t\le T}$, an estimator of $\mathbf{x}$ needs to be derived from an evaluation of the consequences of wrong allocations.

Two common ways corresponding to two loss functions

$$L_1(\mathbf{x},\widehat{\mathbf{x}}) = \sum_{i\in\mathcal{I}} \mathbb{I}_{x_i\neq\hat{x}_i}\,,$$

$$L_2(\mathbf{x},\widehat{\mathbf{x}}) = \mathbb{I}_{\mathbf{x}\neq\widehat{\mathbf{x}}}\,,$$

Corresponding estimators

$$\hat{x}_i^{MPM} = \arg\max_{1\le g\le G} \mathbb{P}^{\pi}(x_i = g|\mathbf{y})\,, \quad i\in\mathcal{I}\,,$$

$$\widehat{\mathbf{x}}^{MAP} = \arg\max_{\mathbf{x}} \pi(\mathbf{x}|\mathbf{y})\,,$$

Image segmentation (2)

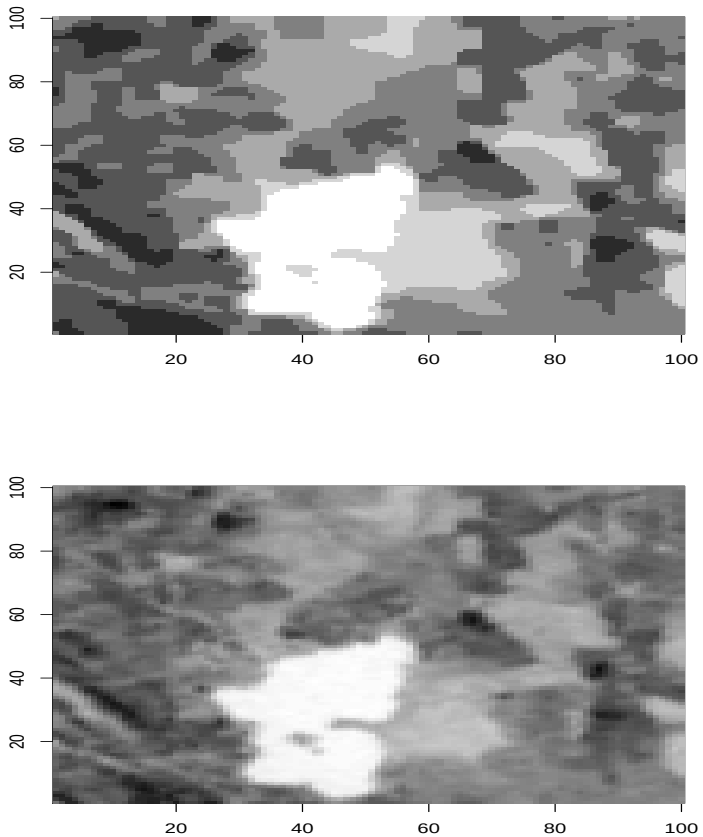
$\widehat{\mathbf{x}}^{MPM}$ and $\widehat{\mathbf{x}}^{MAP}$ not available in closed form!

Approximation

$$\hat{x}_i^{MPM} = \max_{g \in \{1,\ldots,G\}} \sum_{j=1}^{N} \mathbb{I}_{x_i^{(j)}=g} \,,$$

based on a simulated sequence, $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}$.

Image segmentation (3)



(*top*) Segmented image based on the MPM estimate produced after 2000 iterations of the Gibbs sampler and (*bottom*) the observed image