

# STAT545 HW4

Yi Yang

10/3/2018

## 1. Problem 1: Exponential family distributions

1. Consider a random variable  $x$  that can take  $D$  values and that is distributed according to the discrete distribution with parameters  $\vec{\pi}$ . We will write this as  $p(x|\vec{\pi})$ , with  $p(x=c|\vec{\pi}) = \pi_c$  for  $c \in \{1, \dots, D\}$ .
  - (a) Write  $p(x|\vec{\pi})$  as an exponential family distribution and give the natural parameters  $\vec{\eta}$  as a function of  $\pi$  (note this means you can also write  $\pi$  as a function of  $\eta$  though you don't have to). Also write a minimal feature vector  $\phi$  (note  $\pi_D = 1 - \sum_{i=1}^{D-1} \pi_i$ ).
  - (b) Write  $E[\phi(x)]$ , the expectation of the feature vector  $\phi$  as a function of the natural parameters  $\vec{\eta}$ . Recall that given some data  $X = (x_1, \dots, x_N)$ , maximum likelihood estimation (MLE) of  $\eta$  (and thus  $\pi$ ) is moment matching (i.e. calculating the empirical average of  $\pi$  and setting  $\eta$  so that the population average and the empirical average match).

**Solution:**

1. For this discrete distribution, the exponential form is given as follows

(a)

$$\begin{aligned} p(x|\vec{\pi}) &= \prod_{k=1}^D \pi_k^{\delta(x=k)} \\ &= \exp\left(\sum_{k=1}^D \delta(x=k) \cdot \log(\pi_k)\right) \end{aligned}$$

The feature vector is given by  $\phi(x) = [\delta(x=1), \delta(x=2), \dots, \delta(x=D)]^\top$ . The natural parameter is given by  $\eta = [\log(\pi_1), \log(\pi_2), \dots, \log(\pi_D)]^\top$ . The minimal exponential form is derived as follows, the minimal feature vector is given by  $\phi(x) = [\delta(x=1), \delta(x=2), \dots, \delta(x=D-1)]^\top$ .

$$\begin{aligned} p(x|\vec{\pi}) &= \exp\left(\sum_{k=1}^D \delta(x=k) \cdot \log(\pi_k)\right) \\ &= \exp\left(\sum_{k=1}^{D-1} \delta(x=k) \log\left(\frac{\pi_k}{\pi_D}\right) + \sum_{k=1}^{D-1} \delta(x=k) \log(\pi_D) + \delta(x=D) \log(\pi_D)\right) \\ &= \pi_D \exp\left(\sum_{k=1}^{D-1} \delta(x=k) \log\left(\frac{\pi_k}{\pi_D}\right)\right) \end{aligned}$$

- (b) The expectation of the feature vector is given by

$$\mathbb{E}[\phi(x)] = \begin{bmatrix} \mathbb{E}[\delta(x=1)] \\ \vdots \\ \mathbb{E}[\delta(x=D)] \end{bmatrix} = \begin{bmatrix} \pi_1 \\ \vdots \\ \pi_D \end{bmatrix} = \exp(\eta)$$

2. Let  $x$  be Poisson distributed with mean  $\lambda$ . Repeat parts (a), (b).
3. Let  $x$  be a 1-dimensional Gaussian with mean  $\mu$  and variance  $\sigma^2$ . Repeat parts (a), (b) (Note: both  $\mu$  and  $\sigma^2$  are parameters).
4. Let  $x$  follow a geometric distribution with success probability  $p$ : ( $Pr(X=k) = (1-p)^k p$  for  $k = 0, 1, 2, \dots$ ). Repeat parts (a), (b).

**Solution:**

2. The Poisson distribution can be expressed in exponential form as follows

$$\begin{aligned} p(x|\lambda) &= \frac{\lambda^x \exp(-\lambda)}{x!} \\ &= \exp(-\lambda) \frac{1}{x!} \exp(\log(\lambda)x) \end{aligned}$$

The feature vector and the minimal feature vector are both  $\phi(x) = x$ , the natural parameter is  $\eta = \log(\lambda)$ . The expectation of the feature vector is derived as follows

$$\mathbb{E}[\phi(x)] = \mathbb{E}[x] = \sum_{x=1}^{\infty} \frac{\lambda^x \exp(-\lambda)}{(x-1)!} = \lambda = \exp(\eta)$$

3. The 1-D normal distribution can be expressed in exponential form as follows,

$$\begin{aligned} p(x|\mu, \sigma^2) &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) \\ &= \frac{\exp\left(-\frac{\mu^2}{2\sigma^2}\right)}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}x^2 + \frac{\mu}{\sigma^2}x\right) \end{aligned}$$

The feature vector and the minimal feature vector are both  $\phi(x) = [x^2, x]^\top$ , the natural parameter is  $\eta = [-\frac{1}{2\sigma^2}, \frac{\mu}{\sigma^2}]^\top$ . The expectation of the feature vector is given by

$$\mathbb{E}[\phi(x)] = \begin{bmatrix} \mathbb{E}[x^2] \\ \mathbb{E}[x] \end{bmatrix} = \begin{bmatrix} \mu^2 + \sigma^2 \\ \mu \end{bmatrix} = \begin{bmatrix} -\frac{1}{2\eta_1} + \frac{\eta_2^2}{4\eta_1^2} \\ -\frac{1}{2\eta_1} \end{bmatrix}$$

4. The geometric distribution can be expressed in exponential form as follows,

$$\begin{aligned} Pr(X = k) &= (1-p)^X p \\ &= \exp(X \log(1-p) + \log(p)) \end{aligned}$$

The feature vector and the minimal feature vector are both  $x$ , and the natural parameter is  $\log(1-p)$ . The expectation of the feature vector is given by

$$\mathbb{E}[\phi(x)] = \mathbb{E}[x] = \sum_{k=0}^{\infty} k(1-p)^k p = \frac{1-p}{p} = \frac{\exp(\eta)}{1 - \exp(\eta)}$$

**2. Problem 2: EM for mixture of Bernoulli vectors**

1. We looked at the MNIST dataset last assignment. Write code to create a new dataset of only twos and threes using the information in labels. Each pixel can take values from 1 to 256: now threshold the images to be binary (0 or 1). Use a threshold between 1 to 5 (whatever you think is best). Do not use a for loop.

We will model these binary images as a mixture of  $K$  Bernoulli vectors. Thus, we have  $K$  clusters, each of which is parametrized by a 784-dimensional vector with each component lying between 0 and 1. Call the  $k$ th cluster parameter  $\mu^k$ , with  $\mu^k \in [0, 1]^{784}$ . The probability over clusters is a  $k$ -component probability vector  $\pi$ . Thus, to generate an observation, we first sample a cluster  $c$  from  $\pi$ , and then generate a random binary image  $x$  by setting the  $i$ th pixel to 1 with probability  $\mu_i^k$  for  $i$  from 1 to 784.

2. Given  $N$  observations  $X = (x_1, \dots, x_N)$  and their cluster assignments  $C = (c_1, \dots, c_N)$ , write down the log joint-probability  $\log p(X, C|\pi, \vec{\mu})$ .

3. If we observed both  $X$  and  $C$ , what are the maximum likelihood estimates of  $\pi$  and the  $\mu^k$ s?
4. Explain why  $p(C|X, \pi, \vec{\mu}) = \prod_{i=1}^N p(c_i|x_i, \pi, \vec{\mu})$ . Write down  $p(c_i|x_i, \pi, \vec{\mu})$ . This is the  $q$  of the EM algorithm.
5. Write down the variational lower bound  $\mathcal{F}(q, \pi, \vec{\mu})$  for the EM algorithm. Use the first expression in the slides involving the entropy  $H(q)$ .
6. For a given  $q$ , what are the  $\pi$  and  $\vec{\mu}$  that maximize this? These expressions should be a simple relaxation of part (3).
7. Write an EM algorithm that maximizes  $\mathcal{F}$  by alternately maximizing w.r.t.  $q$  (step 4) and  $(\pi, \vec{\mu})$  (step 6). Although the algorithm doesn't require you to evaluate  $\mathcal{F}$ , your code should do this after each update. This is a useful diagnostic for debugging since  $\mathcal{F}$  should never decrease. Your stopping criteria should be when the value of  $\mathcal{F}$  stabilizes.
8. Run the EM algorithm on the binary digits data set for  $K = 2$  and 3. Plot the cluster parameters using `show_digit`. Also plot the trace of the evolution of  $\mathcal{F}$ . Write down the final value of  $\pi$  and  $\mathcal{F}$ . What are the units of the latter?
9. The entropy of a distribution is a measure of how 'random' it is. For  $K = 2$ , calculate the entropy of the final  $q(c_i|x_i, \vec{\mu}, \pi)$  of each digit, and plot the digit with the largest entropy. This is the digit with largest ambiguity about its correct cluster.

**Solution:**

1.

```
# Credit to https://gist.github.com/brendano/39760
# Load the MNIST digit recognition dataset into R

load_image_file <- function(filename) {
  ret = list()
  f = file(filename, 'rb')
  readBin(f, 'integer', n=1, size=4, endian='big')
  ret$n = readBin(f, 'integer', n=1, size=4, endian='big')
  nrow = readBin(f, 'integer', n=1, size=4, endian='big')
  ncol = readBin(f, 'integer', n=1, size=4, endian='big')
  x = readBin(f, 'integer', n=ret$n*nrow*ncol, size=1, signed=F)
  ret$x = matrix(x, ncol=nrow*ncol, byrow=T)
  close(f)
  ret
}

load_label_file <- function(filename) {
  f = file(filename, 'rb')
  readBin(f, 'integer', n=1, size=4, endian='big')
  n = readBin(f, 'integer', n=1, size=4, endian='big')
  y = readBin(f, 'integer', n=n, size=1, signed=F)
  close(f)
  y
}

show_digit <- function(arr784, col=gray(12:1/12), ...) {
  image(matrix(arr784, nrow=28)[,28:1], col=col, axes=F, ...)
}

# load the first 1000 digits and their labels
digits.list <- load_image_file('../HW3/data/train-images-idx3-ubyte')
```

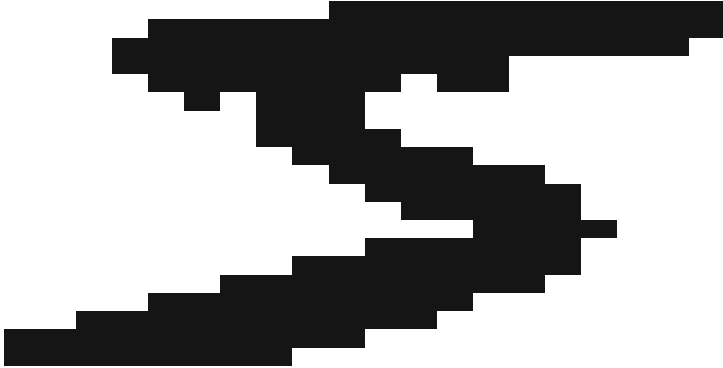
```

digits <- digits.list$x[1:1000,]
#digits <- digits / 255.0 # normalize pixels
digits <- ifelse(digits<4, 0, 1)
labels <- load_label_file('./HW3/data/train-labels-idx1-ubyte')
dim(digits)

```

```
## [1] 1000 784
```

```
show_digit(digits[1,])
```



2. The joint probability is given by

$$p(X, C | \pi, \vec{\mu}) = \prod_{n=1}^N \prod_{k=1}^K \left[ \pi_k \prod_{d=1}^D (\mu_d^k)^{x_{nd}} (1 - \mu_d^k)^{1-x_{nd}} \right]^{\delta(c_n=k)}$$

$$\log p(X, C | \pi, \vec{\mu}) = \sum_{n=1}^N \sum_{k=1}^K \delta(c_n = k) \left\{ \log \pi_k + \sum_{d=1}^D [x_{nd} \log \mu_d^k + (1 - x_{nd}) \log(1 - \mu_d^k)] \right\}$$

3. Consider this constrained optimization problem, introduce the Lagrange multiplier  $\lambda$ , we have:

$$\arg \max \log p(X, C | \pi, \vec{\mu}) + \lambda \left( \sum_{k=1}^K \pi_k - 1 \right)$$

Taking derivative to  $\pi_k$ , we have

$$\frac{\sum_{n=1}^N \delta(c_n = k)}{\pi_k} + \lambda = 0$$

Multiply both sides by  $\pi_k$  and sum over  $k = 1, \dots, K$ , we have

$$\lambda = - \sum_{k=1}^K \sum_{n=1}^N \delta(c_n = k) = -N$$

Therefore, we have

$$\pi_k = \frac{\sum_{n=1}^N \delta(c_n = k)}{N}$$

Taking derivative to  $\mu_d^k$ , we have

$$\sum_{n=1}^N \delta(c_n = k) \left[ \frac{x_{nd}}{\mu_d^k} - \frac{1 - x_{nd}}{1 - \mu_d^k} \right] = 0$$

Therefore, we have

$$\mu_d^k = \frac{\sum_{n=1}^N x_{nd} \delta(c_n = k)}{\sum_{n=1}^N \delta(c_n = k)}$$

4. Each observation is independent to each other. The joint posterior probability is the product of the marginal posterior probability.

$$q_{n,k} = p(c_n = k | x_n, \pi, \vec{\mu}) = \frac{\pi_k p(x_n | \mu^k)}{\sum_{j=1}^K \pi_j p(x_n | \mu^j)}$$

5. The variational lower bound is given by

$$\begin{aligned} F(q, \pi, \vec{\mu}) &= \mathbb{E}_q[\log p(X, C | \pi, \vec{\mu})] + H(q) \\ &= \sum_{n=1}^N \sum_{k=1}^K q_{n,k} \left\{ \log \pi_k + \sum_{d=1}^D [x_{nd} \log \mu_d^k + (1 - x_{nd}) \log(1 - \mu_d^k)] \right\} - \sum_{n=1}^N \sum_{k=1}^K q_{n,k} \log q_{n,k} \end{aligned}$$

6. Analogous to part 3, the estimated parameters should be, given  $N_k = \sum_{n=1}^N q_{n,k}$

$$\begin{aligned} \pi_k &= \frac{N_k}{N} \\ \mu_d^k &= \frac{\sum_{n=1}^N x_{nd} \cdot q_{n,k}}{N_k} \end{aligned}$$

7. The algorithm is implemented as follows

```
em <- function(Obs, K){
  # slog ignore zero logarithm
  slog <- function(val){
    pos <- val > 0
    val[pos] <- log(val[pos])
    return(val)
  }
  set.seed(12)
  # obtain sizes
  N <- nrow(Obs)
  D <- ncol(Obs)
  Pi <- runif(K,0,1)
  Pi <- Pi / sum(Pi)
  mu <- matrix(runif(D*K,0,1),nrow = D)
  q <- matrix(rep(0,N*K), nrow = N)
  temp <- rep(0,K)

  Tol <- 1e-2
  Change <- Tol + 1
  maxIter <- 100
  it <- 0
  Fvalue <- c()
  while (it <= maxIter && Tol <= Change) {
    # E step
    for (n in seq(1,N)){
      x <- Obs[n,]
      for (k in seq(1,K)){
        temp[k] = slog(Pi[k]) + sum(x*slog(mu[,k])) + sum((1 - x)*slog(1 - mu[,k]))
      }
      temp <- temp - max(temp)
      q[n,] <- exp(temp)/sum(exp(temp))
    }
  }
```

```

# M step
Nk <- colSums(q)
Pi <- Nk/N

mu_update <- matrix(rep(0,D*K), nrow = D)
for (kk in seq(1,K)){
  mu_update[,kk] <- colSums(q[,kk]*Obs)/Nk[kk]
}
Change <- max(abs(mu - mu_update))
mu <- mu_update
it <- it + 1

# update F value
fvalue <- 0
for (n in seq(1,N)){
  x <- Obs[n,]
  for (k in seq(1,K)){
    temp[k] = slog(Pi[k]) + sum(x*slog(mu[,k])) + sum((1 - x)*slog(1 - mu[,k]))
    fvalue <- fvalue + q[n,k]*temp[k] - q[n,k]*slog(q[n,k])
  }
}
Fvalue <- c(Fvalue, fvalue)
}

return(list(Pi=Pi,mu=mu,q=q,Fvalue=Fvalue,it=it))
}

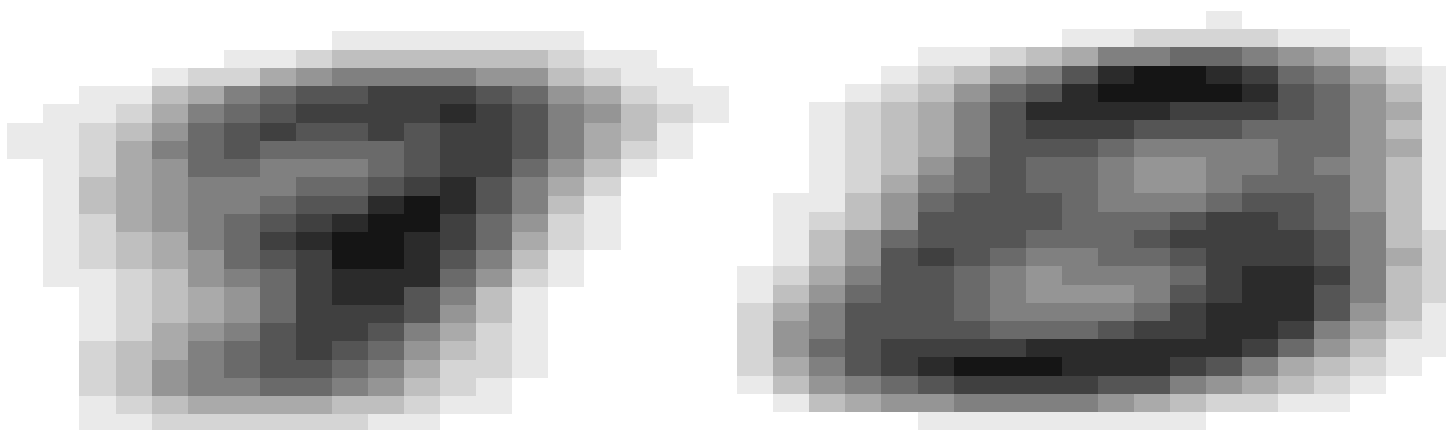
```

8.

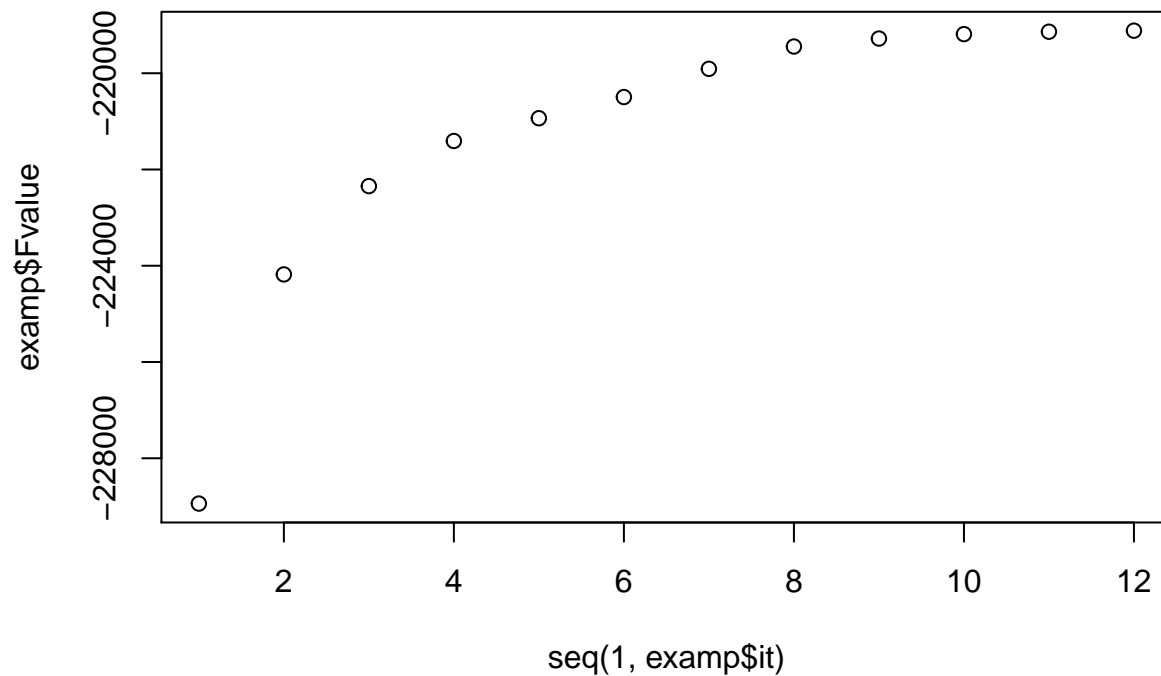
```

#K = 2
examp <- em(digits,2)
for (id in seq(1,ncol(examp$mu))){
  show_digit(examp$mu[,id])
}

```



```
plot(seq(1,examp$it),examp$Fvalue)
```



```
cat(paste('Final value of pi and F:\n'))
```

```
## Final value of pi and F:
```

```
print(examp$Pi)
```

```
## [1] 0.5758176 0.4241824
```

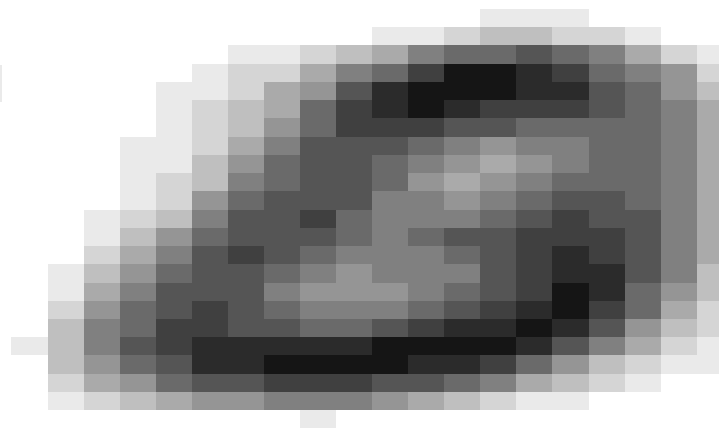
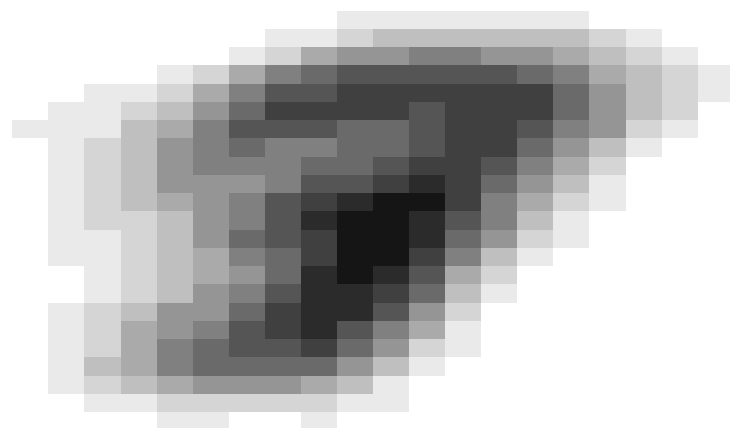
```
print(examp$Fvalue[examp$it])
```

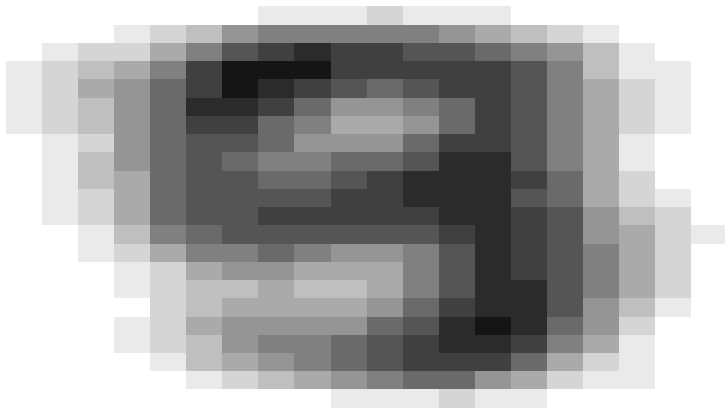
```
## [1] -219116.1
```

```
#K = 3
```

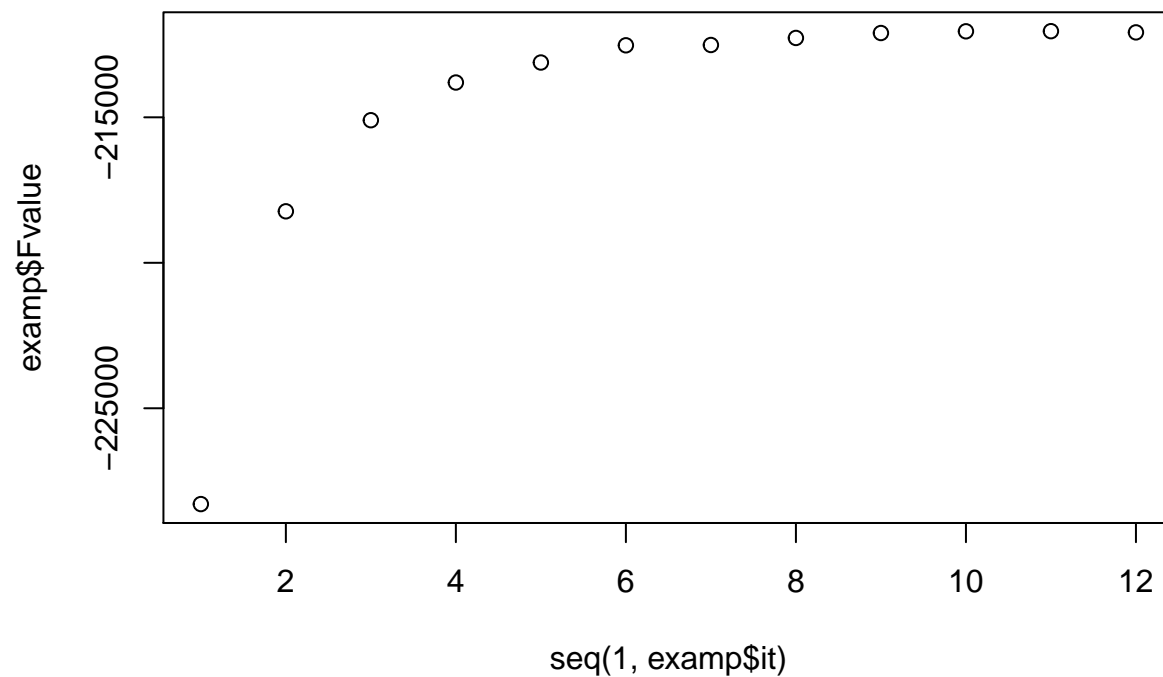
```
examp <- em(digits,3)
```

```
for (id in seq(1,ncol(examp$mu))){
  show_digit(examp$mu[,id])
}
```





```
plot(seq(1,examp$it),examp$Fvalue)
```



```
cat(paste('Final value of pi and F:\n'))
```

```
## Final value of pi and F:
```

```
print(examp$Pi)
```

```
## [1] 0.4569846 0.3563021 0.1867133
```

```
print(examp$Fvalue[examp$it])
```

```
## [1] -212080.2
```

The units of F is the same as the entropy, which is J/K.

9.

```
examp <- em(digits,2)
```

```
Q <- examp$q
```

```
dim(Q)
```

```
## [1] 1000 2
```



```
# slog ignore zero logarithm
slog <- function(val){
  pos <- val > 0
  val[pos] <- log(val[pos])
  return(val)
}
Entropy <- apply(Q,1,function(x) sum(-x*slog(x)))
show_digit(digits[which(Entropy == max(Entropy)),])
```

