```
function f_best = decodemessage(x,M,N)

AZ = 27;                        % Number of characters of interest
scale_factor = 1850;            % Probability scale factor

%
% Initialize data arrays
%
F = zeros(N,AZ);
Pl = zeros(N,1);
p_accept = ones(N,1);

%
% Convert the text to the range 1−27
%
txt = mod(bitand(uint8(x),31),AZ)+1;
L = length(txt);
fprintf(1,'%4d: %s\n',0,char(txt(:).'+95));

%
% Start with a random permutation of the characters
%
F(1,:) = randperm(AZ);

%
% Determine the plausibility of the initial mapping
%
p = 0
for c = 2:length(txt)
    Pl(1) = Pl(1) + M(F(1,(txt(c−1))),F(1,(txt(c))));
end
Pl(1) = Pl(1)/length(txt);

%
% Store the most plausible mapping seen so far
%
Pl_best = Pl(1);
f_best = F(1,:);

%
% Attempt N character swaps in the mapping, favoring the most plausible
%
for iter = 2:N
    %
    % Randomly swap two characters in substitution map
    %
    f = F(iter−1,:);
    ij = randperm(AZ,2);
    f(ij) = f(fliplr(ij));

    %
    % Test the plausibility of the new substitution map
    %
    p = 0;
    for c = 2:length(txt)
        p = p + M(f(txt(c−1)),f(txt(c)));
    end
    p = p/length(txt);

    %
    % If the new substitution map is more plausible, then accept
    %  it. If not, accept it with a probability determined by
    %  the difference between them times a scale factor
    %
    if ( rand() < exp(scale_factor*(p−Pl(iter−1))) )
```

```
        F(iter,:) = f;
        Pl(iter) = p;
        %
        % Test if this is the most plausible mapping we've seen
        %
        if ( p > Pl_best )
            Pl_best = p;
            f_best = f;
            new_txt = char(f(txt(:)).'+95);
            fprintf(1,'%4d: %s\n',iter,new_txt);
        end
    else
        % Keep the old mapping
        F(iter,:) = F(iter−1,:);
        Pl(iter) = Pl(iter−1);
    end
end

%
% Plot and label the transition probability matrix
%
figure(1); clf;
imagesc(M); axis image;
colormap(gray(256)); colorbar;
title('Letter Transition Probabilities (Log Scale)','FontSize',16);
xlabel('To Letter','FontSize',14);
ylabel('From Letter','FontSize',14);
% Set the arrays used for the plot labels
Tick = 1:27;
tl =char(Tick.'+63);
tl(1) = char('_');
TickLabel = mat2cell(tl,length(Tick),1);
% Show the letters along the axis of the image
set(gca,'TickLength',[0 0]);
set(gca,'XTick',Tick);
set(gca,'XTickLabel',TickLabel);
set(gca,'YTick',Tick);
set(gca,'YTickLabel',TickLabel);

%
% Plot the plausibility over time
%
figure(2); clf;
semilogx(Pl);
title('Plausibility Over Time','FontSize',16);
xlabel('Iteration Number','FontSize',14);
ylabel('Log Plausibility','FontSize',14);

end
```

```
#!/bin/bash

function show_help()
{
  echo "Usage: ‘basename $0‘ [−f FILE] [[−d FOLDER] | [−o OUT]]"
  echo "       ‘basename $0‘ [−n NUM] [−m URL] [−d FOLDER]"
  echo "       ‘basename $0‘ [−b BOOK] [−m URL] [[−d FOLDER] | [−o OUT]]"
  echo "Retrieve book(s) from Project Gutenberg catalog and strip the file"
  echo "of all characters except A−Z and <space>."
  echo
  echo "  −f FILE        open and partse the book found at FILE"
  echo "                   (this will ignore options −b −n −m)"
  echo "  −b BOOK        download and parse book number BOOK from PG catalog"
  echo "                   (this will ignore option −n)"
  echo "  −n NUM         download NUM random books (default 1)"
  echo "  −d FOLDER      save results to FOLDER (default \"./words/\")"
  echo "  −o FILE        save results to output file FILE"
  echo "                   (do not use with option −n)"
  echo "  −m URL         retrieve books from mirror at URL"
  echo "                   (default \"ftp://mirrors.xmission.com/gutenberg\")"
  echo "  −u             do not strip header and footer"
  echo "  −l             language−agnostic download"
  echo "  −v             show verbose output (for debugging)"
  echo "  −h, −−help     display this message and exit"
  echo
  echo "  Report bugs to josh+git@nispio.net"
}

# Check for −−help option
if [ "$1" = "−−help" ]; then show_help; exit 0; fi

# Set default values
MIRROR="ftp://mirrors.xmission.com/gutenberg/"
BOOK_FOLDER="./books/"
OUTPUT_FOLDER="./words/"
TARGET_PATH=""
ITERS=10
BOOK_NUM=0
STRIP_HEADERS=1
LANG="ENGLISH"

# NOTE: The Project Gutenberg Terms of Service state that the site is intended
# for human users only. For that reason, this script uses a mirror site rather
# than downloading directly from gutenberg.org.  For more info see:
#
#  www.gutenberg.org/wiki/Gutenberg:Information_About_Robot_Access_to_our_Pages
#

OPTIND=1                        # Reset in case getopts has been used previously
opts="hvlf:b:n:d:o:m:"          # Options for call to getopts
# Parse input arguments
while getopts "$opts" opt; do
    case "$opt" in
        h)  show_help; exit 0 ;;
        v)  set −x ;;
        l)  LANG="" ;;
        u)  STRIP_HEADERS="" ;;
        f)  BOOK_FILE="$OPTARG" ;;
        n)  ITERS="$OPTARG" ;;
        b)  BOOK_NUM="$OPTARG"; ITERS=1 ;;
        d)  OUTPUT_FOLDER="$OPTARG" ;;
        o)  BOOK_FOLDER="$OPTARG" ;;
        m)  MIRROR="$OPTARG" ;;
        \?) exit 1 ;;
        :)  echo "Option −$OPTARG requires an argument"; exit 1 ;;
```

```
    esac
done

[ ! −d "$BOOK_FOLDER" ]   &&  mkdir "$BOOK_FOLDER"
[ ! −d "$OUTPUT_FOLDER" ] &&  mkdir "$OUTPUT_FOLDER"

# Function that strips the headers/footers and punctuation
function strip_book()
{
    BOOK="$1"
    OUTFILE="$2"

    # Make sure the book is in the selected language
    book_lang=$(grep −m 1 −i −E '^Language:' "$BOOK" | cut −f2 −d' ' )
    if [ −n "$LANG" ] && [ −z "$book_lang" ]
    then
        echo "This book's language could not be determined. Skipping."
        echo;
        return 1;
    elif [ ! $LANG=$(echo "$book_lang" | tr 'a−z' 'A−Z' ) ]
    then
        echo "This book is not in English. Language is $book_lang. Skipping."
        echo;
        return 1;
    fi

    # Find where the book actually begins and ends
    #
    #   Each Project Gutenberg book has a header and footer with copyright
    #   and other information.  Since this text is (nearly) identical in
    #   every book, it would introduce a bias in our training data.  For
    #   that reason, we trim it out of the file. We also discard some
    #   pre−determined number of lines at the beginning and end of the book,
    #   since these lines often contain tables of contents and indices whose
    #   content is atypical of English prose.
    #
    if [ −n "$STRIP_HEADERS" ]
    then
        HEADER_START="START OF (THE|THIS) PROJECT GUTENBERG EBOOK"
        FOOTER_START="END OF (THE|THIS) PROJECT GUTENBERG EBOOK"
        bookstart=$(grep −n −m 1 −i −E "$HEADER_START" "$BOOK" | cut −f1 −d':')
        bookend=$(grep −n −m 1 −i −E "$FOOTER_START" "$BOOK" | cut −f1 −d':')
        prescript=100
        postscript=100
        total_lines=$(wc −l "$BOOK" | cut −f1 −d' ')

        # If the header or footer were not found, don't worry about it
        [ −z "$bookstart" ] && bookstart=0
        [ −z "$bookend" ] && bookend=0

        # Trim Gutenberg headers and footers and remove DOS line endings
        t=$(($total_lines − $bookstart − $prescript))
        h=$(($bookend − $bookstart − $prescript − $postscript))
        tail −n $t "$BOOK" | head −n $h | tr −d '\r' > "$OUTFILE"
    else
        cat "$BOOK" |  tr −d '\r' > "$OUTFILE"
    fi

    # Use sed to achieve the following (in order):
    #   Set a label
    #   Read in the next line
    #   If we aren't at the end of the file go back to label 'a'
    #   Replace line endings with a space
    #   Convert all letters to upper−case
    #   Trim apostrophes (otherwise we see lots of orphaned s chars
```

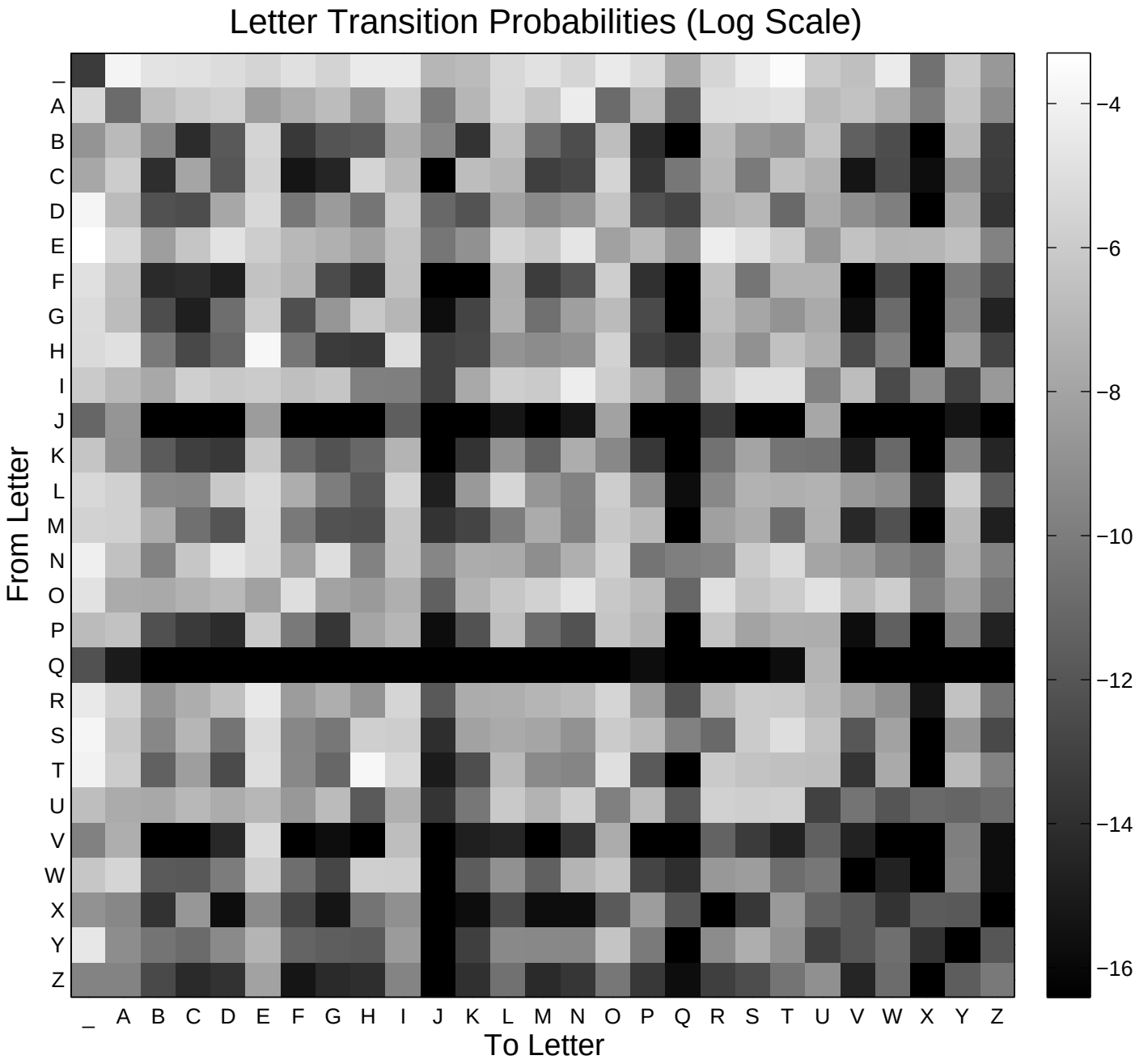
```
    #   Throw out any characters that aren't a−z or <space>
    #   Convert any amount of whitespace into a single @ character
    sed −i \
        −e ':a' \
        −e 'N' \
        −e '$!ba' \
        −e 'y/\n/ /' \
        −e 'y/abcdefghijklmnopqrstuvwxyz/ABCDEFGHIJKLMNOPQRSTUVWXYZ/' \
        −e 's/\([A−Z]\)'"'"'\([A−Z]\)/\1\2/g' \
        −e 's/[^A−Z ]/ /g' \
        −e 's/\s\+/@/g' \
        "$OUTFILE"

    echo Words saved to file "$OUTFILE"
    echo;
    return 0;
}

# If a specific file was given to process, run the processing and then exit
if [ −r "$BOOK_FILE" ]
then
    filename=$(basename "$BOOK_FILE")
    OUTFILE="$OUTPUT_FOLDER/${filename%.*}.out"
    strip_book "$BOOK_FILE" "$OUTFILE"
    exit $?
elif [ −n "$BOOK_FILE" ]
then
    echo File not found: "$BOOK_FILE"
    exit 1
fi

# Download the number of books specified by the user and process each one
for i in $(seq 1 $ITERS)
do
    # Choose a random number between 10000 and 42767
    #
    #   Project Gutenberg has assigned a unique identifier to every book in
    #   their catalog. These numbers run from 1 to ~44000. We can only generate
    #   random numbers with a range of 32768, so we choose to draw numbers in a
    #   range that guarantess exactly 5 digits in order to simplify later steps.
    #
    let "n = (10000+$RANDOM) %100000"

    # If a specific book number was given, override the random n
    if [ ! $BOOK_NUM −eq 0 ]; then let "n = $BOOK_NUM % 100000"; fi

    # Separate the decimal digits of n
    let "a4 = $n/10000 %10"
    let "a3 = $n/1000  %10"
    let "a2 = $n/100   %10"
    let "a1 = $n/10    %10"
    let "a0 = $n/10    %10"

    # Generate the appropriate URL and filename for the given book number
    BOOK_URL="${MIRROR%%/}/${a4}/${a3}/${a2}/${a1}/${n}/${n}.txt"
    BOOK="$BOOK_FOLDER/${n}.txt"
    OUTFILE="${OUTPUT_FOLDER%%/}/${n}.out"

    [ −n "$TARGET_FILE" ] && OUTFILE="$TARGET_FILE"

    # Fetch the book from the mirror site
    wget "$BOOK_URL" −−output−document="$BOOK" || rm "$BOOK"

    # If the file wasn't found on the server, break here
    if [ ! −e "$BOOK" ]
```

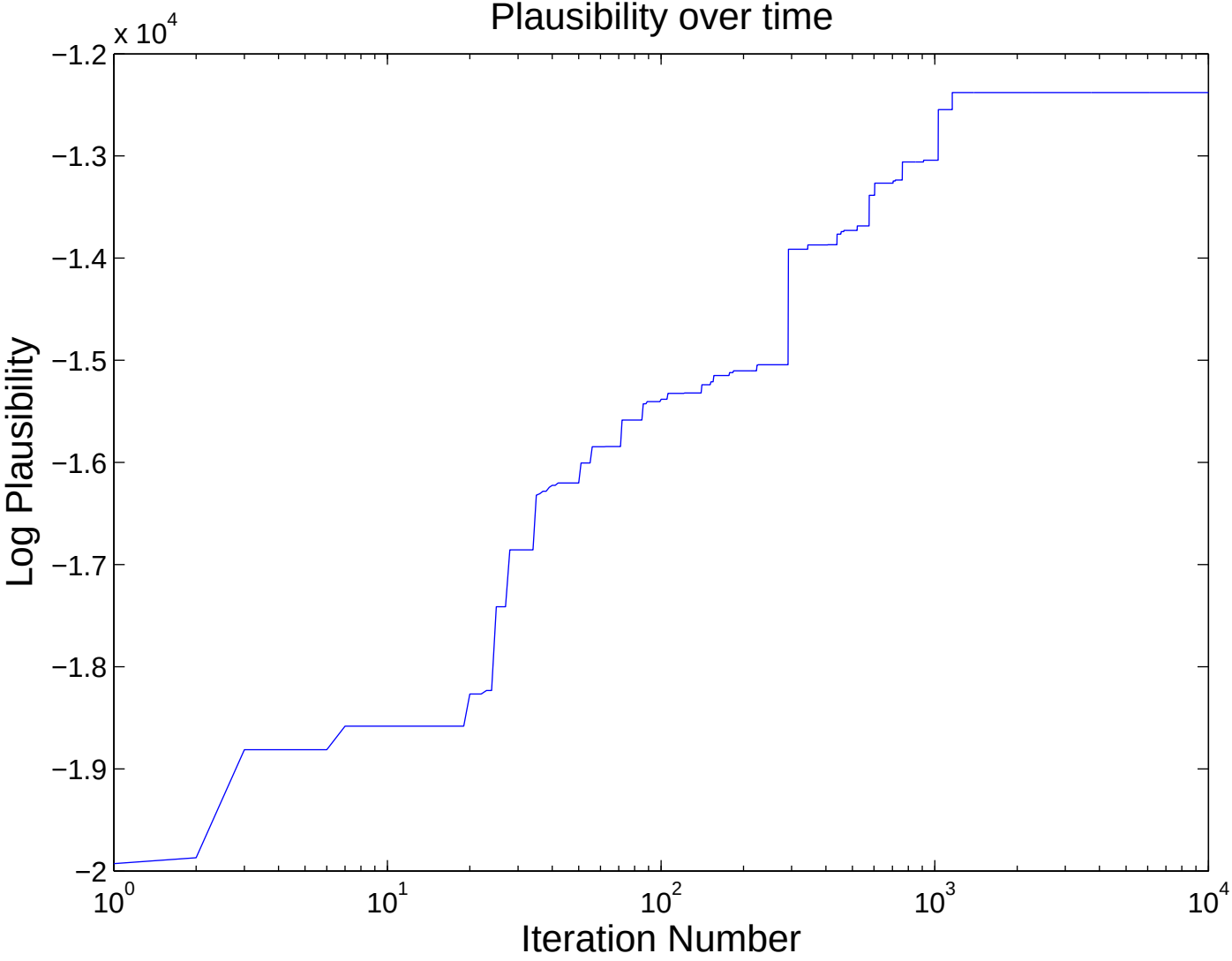
```
    then
        echo "Project Gutenberg book #$n not found. Try again."
        echo;
        continue;
    fi

    strip_book "$BOOK" "$OUTFILE"
done


# Copyright 2013 Josh Hunsaker (josh+git@nispio.net)

# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.

# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.

# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
```

```
function M = parsebooks(folder, name)

A = uint8('A');                         % ASCII representation of A
Z = uint8('Z');                         % ASCII representation of Z
C = Z−(A−1)+1;                          % Number of characters of interest
T = ones(C,C);                          % Initialize transition count matrix

%
% Loop over all of the files in the specified folder
%
filename = dir(fullfile(folder,name));
for book = 1:size(filename,1)
   %
   % Open the file for reading
   %
   filepath = fullfile(folder,filename(book).name);
   fid = fopen(filepath);
   if fid > 0
       display(sprintf('Reading: "%s"', filepath));
       txt = uint8(fread(fid,Inf));
       fclose(fid);
   else
       continue;
   end

   %
   % Shift the characters down to the range 1−27
   %
   L = mod(txt−(A−1),C)+1;

   %
   % Loop over each consecutive pair of letters
   %
   for idx = 2:length(L)
       %
       % Get the current count for this transition
       % and increment the transition count by 1
       %
       t = T(L(idx−1),L(idx));
       T(L(idx−1),L(idx)) = t + 1;
   end
end

%
% Convert transition counts to probabilities
%
M = log(T/sum(sum(T)));

end
```

```
   0: ubj dohwebsjlnbjhbaisilwdojdfjhwqoimhjnirwoqjijhwxambcj dohlsiwoljyihbejdo ...
   2: futywvszlugt autsupege zwvtwktszjverstaebzvjtetszipruhtywvs gezv toesultwv ...
   3: futywvslzugt autsupege lwvtwktsljverstaeblvjtetslipruhtywvs gelv toesuztwv ...
   7: futywvslzurt autsupere lwvtwktsljvegstaeblvjtetslipguhtywvs relv toesuztwv ...
  20: futywvslhurt autsupere lwvtwktsljvegstaeblvjtetslipguztywvs relv toesuhtwv ...
  23: cutywvslhurt autsupere lwvtwktsljvegstaeblvjtetslipguztywvs relv toesuhtwv ...
  25: cutyovslhurt autsupere lovtoktsljvegstaeblvjtetslipguztyovs relv twesuhtov ...
  28: catyovslhart uatsapere lovtoktsljvegstueblvjtetslipgaztyovs relv twesahtov ...
  35: catyomslhart uatsapere lomtoktsljmegstueblmjtetslipgaztyoms relm twesahtom ...
  36: catyomslhart uatsanere lomtoktsljmegstueblmjtetslingaztyoms relm twesahtom ...
  37: catyomslhart uatsaiere lomtoktsljmegstueblmjtetslnigaztyoms relm twesahtom ...
  39: cetyomslhert uetseiara lomtoktsljmagstuablmjtatslnigeztyoms ralm twasehtom ...
  40: cetyomslhert uetseiara lomtoktsljmagstuablmjtatslnigextyoms ralm twasehtom ...
  42: cetyomslhert uetseiara lomtoktsljmagstuadlmjtatslnigextyoms ralm twasehtom ...
  51: cetyomsluert hetseiara lomtoktsljmagsthadlmjtatslnigextyoms ralm twaseutom ...
  56: cetyomsluert hetsegara lomtoktsljmaisthadlmjtatslngiextyoms ralm twaseutom ...
  63: cetyomsluert hetsefara lomtoktsljmaisthadlmjtatslnfiextyoms ralm twaseutom ...
  72: cetyomsluert hetsefara lomtoktslgmaisthadlmgtatslnfiextyoms ralm twaseutom ...
  86: cetyodsluert hetsefara lodtoktslgdaisthamldgtatslnfiextyods rald twaseutod ...
  89: cetyodsluert hetsefara lodtoktslgdaisthazldgtatslnfiextyods rald twaseutod ...
 100: cetyodsluert hetsefara lodtoktslgdaisthazldgtatslnfiextyods rald twaseutod ...
 106: cetyodsluert hetsefara lodtoktslgdaisthavldgtatslnfiextyods rald twaseutod ...
 122: cetyodsluert hetsefara lodtoktslgdaisthavldgtatslnfiebtyods rald twaseutod ...
 141: cetyodsluert hetsemara lodtoktslgdaisthavldgtatslnmiebtyods rald twaseutod ...
 152: cetyodsluert hetsewara lodtoktslgdaisthavldgtatslnwiebtyods rald tmaseutod ...
 156: cetyodsnuert hetsewara nodtoktsngdaisthavndgtatsnlwiebtyods rand tmaseutod ...
 178: wetyodsnuert hetsecara nodtoktsngdaisthavndgtatsnlciebtyods rand tmaseutod ...
 184: wetcodsnuert hetseyara nodtoktsngdaisthavndgtatsnlyiebtcods rand tmaseutod ...
 224: wetcodsnuert hetseyara nodtoftsngdaisthavndgtatsnlyiebtcods rand tmaseutod ...
 226: wetcodsnuert hetsepara nodtoftsngdaisthavndgtatsnlpiebtcods rand tmaseutod ...
 292: we codsnuer the separatnod of sngdais havndg a snlpieb codstrandt maseu od ...
 344: we codsnuer the separatnod of sngdais havndg a snlpiey codstrandt maseu od ...
 407: we codsnuer the separatnod of sngdais havndg a snlpiex codstrandt maseu od ...
 440: we codsuner the separatuod of sugdais havudg a sulpiex codstraudt masen od ...
 455: we codsuner the separatuod of sugdals havudg a suiplex codstraudt masen od ...
 466: we codsuner the seiaratuod of sugdals havudg a supilex codstraudt masen od ...
 521: we codsuner the seiaratuod of sugdals havudg a sumilex codstraudt pasen od ...
 576: we codsiner the seuaratiod of sigdals havidg a simulex codstraidt pasen od ...
 604: we codsiner the separatiod of sigdals havidg a simplex codstraidt uasen od ...
 705: we codsiner the separatiod of sigdals havidg a simplex codstraidt kasen od ...
 718: we codsiner the separatiod of sigdals havidg a simplex codstraidt basen od ...
 762: we codsiner the separatiod of sigdals havidg a simplex codstraidt basen od ...
 910: we codsiner the separatiod of sigdals havidg a simplex codstraidt basen od ...
1030: we consider the separation of signals having a simplex constraint based on ...
1159: we consider the separation of signals having a simplex constraint based on ...
```

we consider the separation of signals having a simplex constraint based on observations of n
oisy data using a fully bayesian result a joint density for the abundances the simplex const
rained signal is established to specifically allow for the abundances to be sparse this join
t density is endowed with a parameter which is parameterized to encourage sparseness unlike
related previous work which did not specifically impose the sparsity assumption overall a gi
bbs sampling framework is established from which all parameters can be learned a metropolis
sampling framework is developed for the abundances mixture coefficients which explicitly and
efficiently represents the sparseness for the parameter of the dirichlet distribution gover
ning sparseness the posterior is shown to be closely approximated by a gamma thus the entire
set of parameters can be efficiently learned by sampling we consider the signal model in wh
ich both and are to be identified from noisy observations and where each is drawn from a pro
bability simplex this model is of interest in hyperspectral unmixing where columns of repres
ent emissivity and the elements are abundances of spectral components appearing in the obser
vation throughout this paper we will use this language of hyperspectral processing in descri
bing components of the model but the constrained model also fits compositional data problems
generally identifying and has been done using independent component analysis blind source s
eparation except that the simplex constraint of violates the fundamental assumption of indep
endence in the components because of its importance the problem has nevertheless been approa
ched by a variety of methods which may be summarized as geometrical statistical and sparse r
egressive of these our method is most similar to while assumes a uniform dirichlet prior in
contrast we allow for a more general dirichlet prior distribution this dirichlet prior consi
derably complicates the posterior distribution but encourages sparseness of which is a physi
cally reasonable assumption since any given pixel is a priori expected to be a mixture of on
ly a few components the parameter governing the dirichlet is in turn governed by a hyperprio
r in a manner analogous to the relevance vector machine so that the model learns the degree
of sparseness from the data


Letter Transition Probabilities (Log Scale)
From Letter
To Letter
_ A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
−4
−6
−8
−10
−12
−14
−16


Plausibility over time
x 10^4
−1.2
−1.3
−1.4
−1.5
−1.6
−1.7
−1.8
−1.9
−2
Log Plausibility
10^0
10^1
10^2
10^3
10^4
Iteration Number