Implementação Rápida de ECVQ

José Gabriel R. C. Gomes

UFRJ / COPPE

CPE718 – Aula #10 – Parte I

Programa Básico

```
clear all; close all; S = 0.01; BKJ = [];

for s = 1:400,

    lambda = S*(s-1);
    randn('state',0); rand('state',0); M = 2; N = 800; K = 8; e = 0.5;
    X = randn(M,N);
    Y = 0.5*randn(M,K);
    l = log2(K)*ones(1,size(Y,2));
    F = 200; BK = zeros(F,4);

    for i=1:F,
        % Partition
        J = 0; for n=1:N, j = sum((repmat(X(:,n),1,size(Y,2)) - Y).^2,1) + lambda*l;
                   k(n) = min(find(j==min(j))); J = J + min(j); end; J = J/N;
        % Centroid
        p = zeros(K,1); Y = zeros(size(Y)); for n=1:N, Y(:,k(n)) = Y(:,k(n)) + X(:,n);
                                                p(k(n)) = p(k(n)) + 1; end;
        for j=1:K, if p(j)~=0, Y(:,j) = Y(:,j)/p(j); end; end;
        % Cost Evaluation
        D = 0; for n=1:N, D = D + sum((X(:,n)-Y(:,k(n))).^2); end; D = D/N;
        Y = Y(:,find(p~=0)); p = p(find(p~=0));
        p = p/sum(p); H = -sum(p.*log2(p)); BK(i,:) = [D H D+lambda*H J];
        % Codeword Length Update
        l = HuffLen(p)';
    end;

    BKJ = [BKJ ; [lambda D H D+lambda*H size(Y,2)]]; [s lambda D H D+lambda*H size(Y,2)]

end;

plot(BKJ(:,3),BKJ(:,2),'k.'); grid on; xlabel('H (bits per vector)'); ylabel('D (MSE)');

save Aula2B;
```

Tempo de Execução

- Programa Básico: 48 minutos
- Programa Básico Modificado (XYtoYP): 28 minutos
- Implementação MSVC / MEX Debug: 35 segundos
- Implementação MSVC / MEX Release: 22 segundos
- Implementação MEX –setup: 23 segundos

Programa Básico

```
clear all; close all; S = 0.01; BKJ = [];

for s = 1:400,

    lambda = S*(s-1);
    randn('state',0); rand('state',0); M = 2; N = 800; K = 8; e = 0.5;
    X = randn(M,N);
    Y = 0.5*randn(M,K);
    l = log2(K)*ones(1,size(Y,2));
    F = 200; BK = zeros(F,4);

    for i=1:F,
        % Partition
        J = 0; for n=1:N, j = sum((repmat(X(:,n),1,size(Y,2)) - Y).^2,1) + lambda*l;
                   k(n) = min(find(j==min(j))); J = J + min(j); end; J = J/N;
        % Centroid
        p = zeros(K,1); Y = zeros(size(Y)); for n=1:N, Y(:,k(n)) = Y(:,k(n)) + X(:,n);
                                                p(k(n)) = p(k(n)) + 1; end;
        for j=1:K, if p(j)~=0, Y(:,j) = Y(:,j)/p(j); end; end;
        % Cost Evaluation
        D = 0; for n=1:N, D = D + sum((X(:,n)-Y(:,k(n))).^2); end; D = D/N;
        Y = Y(:,find(p~=0)); p = p(find(p~=0));
        p = p/sum(p); H = -sum(p.*log2(p)); BK(i,:) = [D H D+lambda*H J];
        % Codeword Length Update
        l = HuffLen(p)';
    end;

    BKJ = [BKJ ; [lambda D H D+lambda*H size(Y,2)]]; [s lambda D H D+lambda*H size(Y,2)]

end;

plot(BKJ(:,3),BKJ(:,2),'k.'); grid on; xlabel('H (bits per vector)'); ylabel('D (MSE)');

save Aula2B;
```

Programa Básico

```
% Partition
J = 0;
for n=1:N,
         j = sum((repmat(X(:,n),1,size(Y,2)) - Y).^2,1) + lambda*l;
         k(n) = min(find(j==min(j)));
         J = J + min(j);
end;
J = J/N;
% Centroid
p = zeros(K,1);
Y = zeros(size(Y));
for n=1:N,
         Y(:,k(n)) = Y(:,k(n)) + X(:,n);
         p(k(n)) = p(k(n)) + 1;
end;
for j=1:K,
         if p(j)~=0, Y(:,j) = Y(:,j)/p(j); end;
end;
% Cost Evaluation
D = 0; for n=1:N, D = D + sum((X(:,n)-Y(:,k(n))).^2); end; D = D/N;
Y = Y(:,find(p~=0)); p = p(find(p~=0));
p = p/sum(p); H = -sum(p.*log2(p)); BK(i,:) = [D H D+lambda*H J];
% Codeword Length Update
l = HuffLen(p)';
```

Programa Básico Modificado (XYltoYp)

```
function [Yout,p] = XYltoYp(X,Yin,l,lambda);

p = zeros(size(Yin,2),1); Y = Yin; Ynew = zeros(size(Y));
for n=1:size(X,2),
         % Partition
         j = sum((repmat(X(:,n),1,size(Y,2)) - Y).^2,1) + lambda*l;
         k = min(find(j==min(j)));
         % Centroid
         Ynew(:,k) = Ynew(:,k) + X(:,n);
         p(k) = p(k) + 1;
end;
for j=1:size(Yin,2),
         if p(j)~=0, Yout(:,j) = Ynew(:,j)/p(j); end;
end;
```

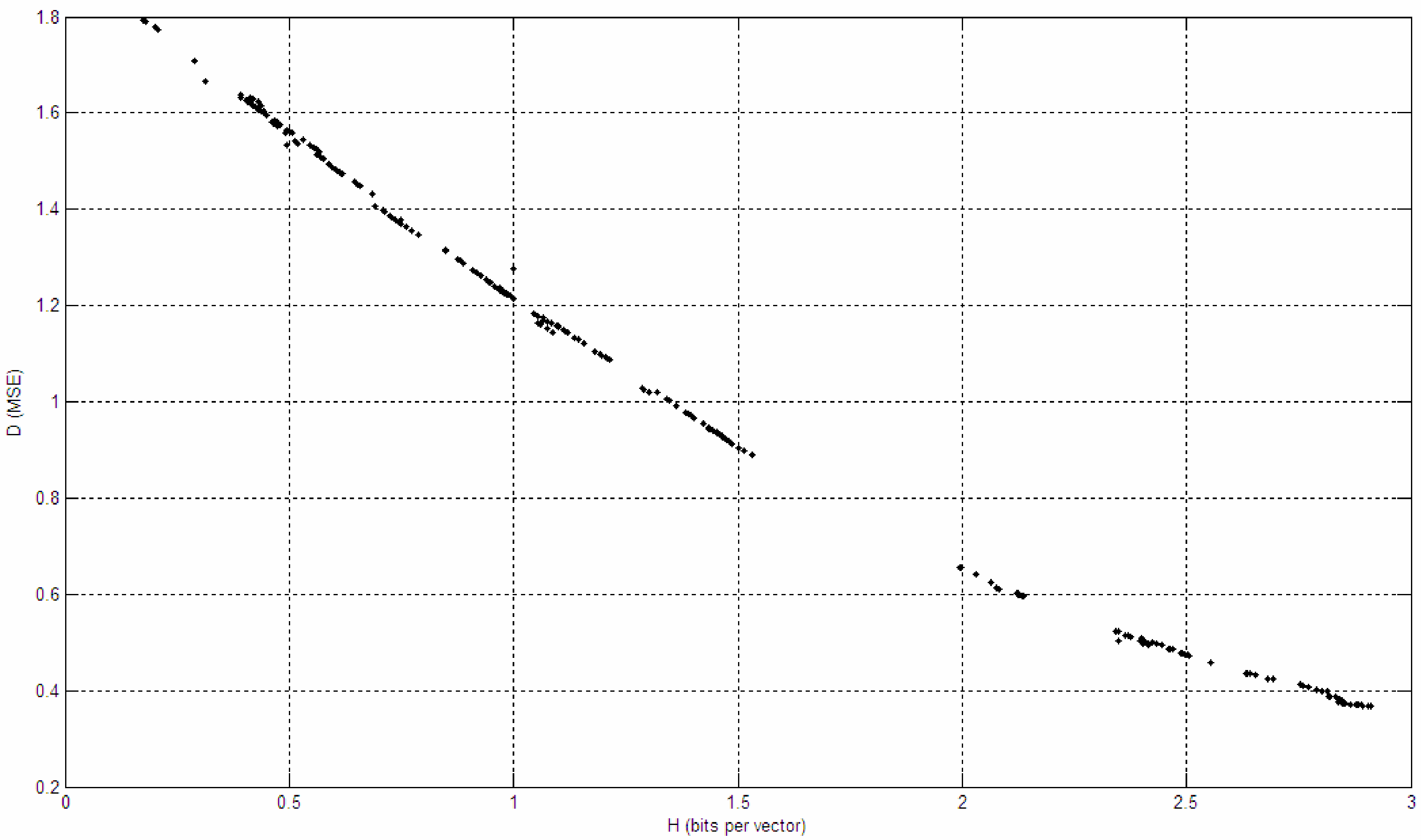
Programa Básico Modificado (XYltoYpD)

```
function [Yout,p,D] = XYltoYpD(X,Yin,l,lambda);

p = zeros(size(Yin,2),1); Y = Yin; Ynew = zeros(size(Y)); D = 0;
for n=1:size(X,2),
         % Partition
         j = sum((repmat(X(:,n),1,size(Y,2)) - Y).^2,1) + lambda*l;
         k = min(find(j==min(j)));
         D = D + sum((X(:,n)-Y(:,k)).^2);
         % Centroid
         Ynew(:,k) = Ynew(:,k) + X(:,n);
         p(k) = p(k) + 1;
end;
D = D/size(X,2);
for j=1:size(Yin,2),
         if p(j)~=0, Yout(:,j) = Ynew(:,j)/p(j); end;
end;
```

Programa Básico Modificado

```
clear all; close all; S = 0.01; BKJ = [];

for s = 1:400,

    lambda = S*(s-1);
    randn('state',0); rand('state',0); M = 2; N = 800; K = 8; e = 0.5;
    X = randn(M,N);
    Y = 0.5*randn(M,K);
    l = log2(K)*ones(1,size(Y,2));
    F = 200; BK = zeros(F,4);

    for i=1:F-1,
        [Y,p] = XYltoYp(X,Y,l,lambda);
        Y = Y(:,find(p~=0));
        p = p(find(p~=0));
        p = p/sum(p);
        % Codeword Length Update
        l = HuffLen(p)';
    end;

    [Y,p,D] = XYltoYpD(X,Y,l,lambda);
    p = p(find(p~=0)); p = p/sum(p);
    H = -sum(p.*log2(p));

    BKJ = [BKJ ; [lambda D H D+lambda*H size(Y,2)]]; [s lambda D H D+lambda*H size(Y,2)]

end;

plot(BKJ(:,3),BKJ(:,2),'k.'); grid on; xlabel('H (bits per vector)'); ylabel('D (MSE)');

save Aula3;
```

Programa Básico Modificado



XYltoYp – Implementação MSVC / MEX

```
// 070614 gabriel@pads.ufrj.br (from 040224 CoreECVQ2.cpp)
// MATLAB Syntax is [Y,p] = XYltoYp_MF(X,Y,l,lambda);

#include <stdlib.h>
#include <math.h>
#include "mex.h"

void mexFunction(int nlhs, mxArray *plhs[], int nrhs, const mxArray *prhs[])

{
              // function [Yout,p] = XYltoYp_MF(X,Yin,l,lambda);

              if (nrhs != 4) mexErrMsgTxt("4 input arguments required.");
              if (nlhs != 2) mexErrMsgTxt("2 output arguments required.");

              // Getting input arguments

              double *X, *Yin, *lambda, *l;
              X = mxGetPr(prhs[0]); Yin = mxGetPr(prhs[1]); l = mxGetPr(prhs[2]); lambda = mxGetPr(prhs[3]);

              // Size definitions

              const int *size_X, *size_Y;
              size_X = mxGetDimensions(prhs[0]); size_Y = mxGetDimensions(prhs[1]);
              int n_dimensions = size_X[0]; int n_elements = size_X[1]; int size_codebook = size_Y[1];

              // Auxiliary stuff

              int i, j, k, data_index, vector_index, codebook_index, best_index, offset;
              double e; double d; double J_min; double J;

              // Getting output arguments

              plhs[0] = mxCreateDoubleMatrix(n_dimensions,size_codebook,mxREAL);
              plhs[1] = mxCreateDoubleMatrix(1,size_codebook,mxREAL);
              double *Yout, *p;
              Yout = mxGetPr(plhs[0]); p = mxGetPr(plhs[1]);

              // Main Code: ECVQ encode (evaluation of new cells and their density) ...
```

XYltoYp – Implementação MSVC / MEX

```
// Main Code: ECVQ encode (evaluation of new cells and their density)

data_index=0; codebook_index=0;

// p = zeros(size(Yin,2),1); Y = Yin; Ynew = zeros(size(Y));

for (i=0 ; i<size_codebook ; i++)
{
               p[i]=0.0;
               for (j=0 ; j<n_dimensions ; j++)
               {
                              Yout[codebook_index]=0.0;
                              codebook_index++;
               }
}

// for n=1:size(X,2), ...
```

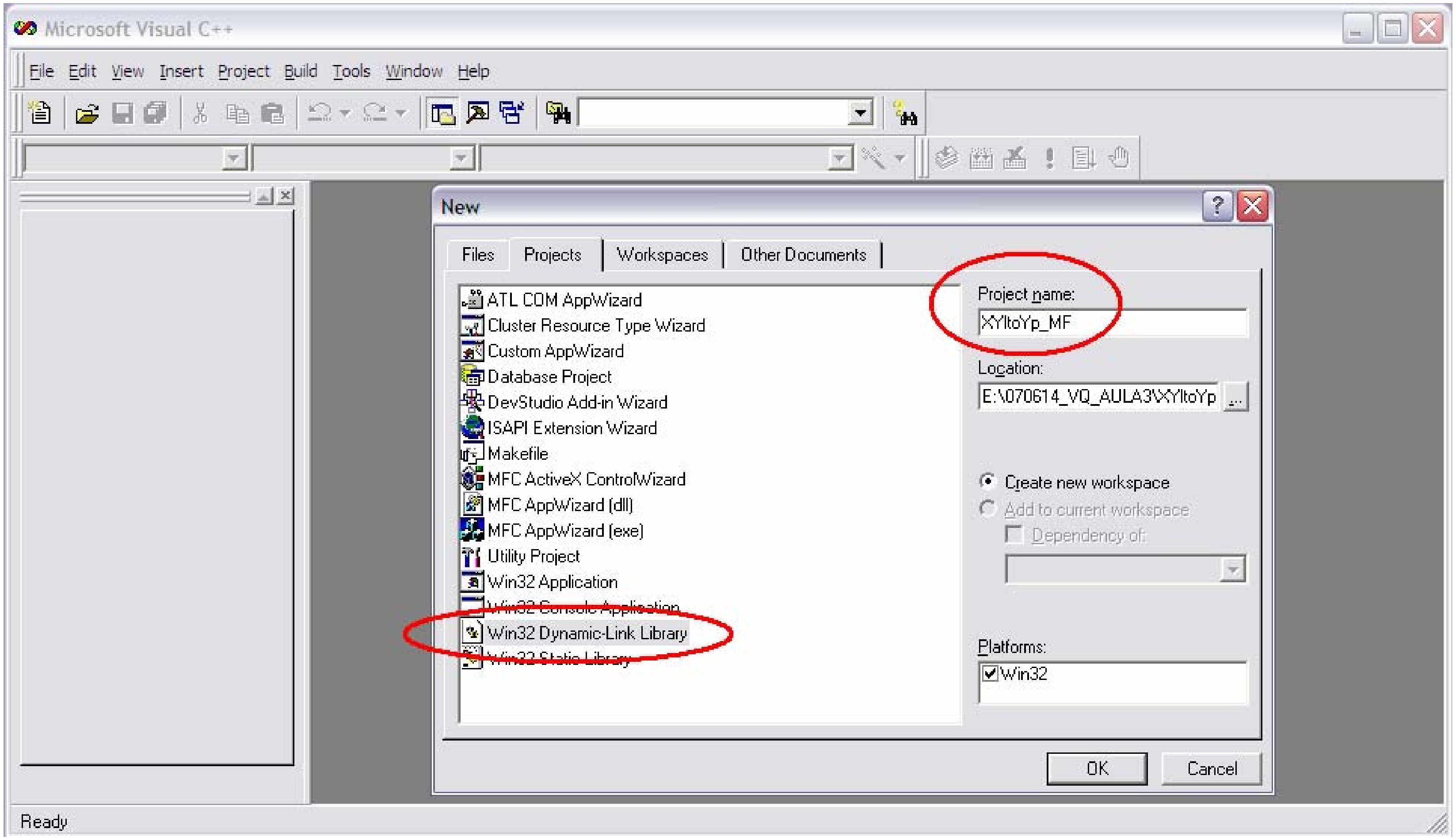
XYltoYp – Implementação MSVC / MEX

```
// for n=1:size(X,2),

for (i=0 ; i<n_elements ; i++)
{
        // j = sum((repmat(X(:,n),1,size(Y,2)) - Y).^2,1) + lambda*l;
        // k = min(find(j==min(j)));

        vector_index=0; codebook_index=0;
        d = 0.0;
        for (k=0 ; k<n_dimensions ; k++)
        {
                e = X[data_index+k]-Yin[codebook_index];
                d = d + e*e;
                codebook_index++;
        }
        J_min = d + (*lambda)*l[vector_index];
        best_index = 0;
        vector_index++;

        for (j=0 ; j<(size_codebook-1) ; j++)
        {
                d = 0.0;
                for (k=0 ; k<n_dimensions ; k++)
                {
                        e = X[data_index+k]-Yin[codebook_index];
                        d = d + e*e;
                        codebook_index++;
                }
                J = d + (*lambda)*l[vector_index];
                if (J < J_min)
                {
                        J_min=J;
                        best_index=vector_index;
                }
                vector_index++;
        }

        // Ynew(:,k) = Ynew(:,k) + X(:,n);
        // p(k) = p(k) + 1;
```

XYltoYp – Implementação MSVC / MEX

```
                              // Ynew(:,k) = Ynew(:,k) + X(:,n);
                              // p(k) = p(k) + 1;

                              offset=best_index*n_dimensions;
                              for (j=0 ; j<n_dimensions ; j++)
                              {
                                             Yout[offset+j] = Yout[offset+j]+X[data_index];
                              data_index++;
                              }
                              p[best_index]=p[best_index]+1.0;
               }

               // for j=1:size(Yin,2),
               //             if p(j)~=0, Yout(:,j) = Ynew(:,j)/p(j); end;
               // end;

               for (i=0 ; i<size_codebook ; i++) if (p[i]!=0) for (j=0 ; j<n_dimensions ; j++)
               Yout[i*n_dimensions+j] = Yout[i*n_dimensions+j]/p[i];
}
```

XYltoYp – Implementação MSVC / MEX



Project name: XYltoYp_MF

Location: E:\070614_VQ_AULA3\XYltoYp

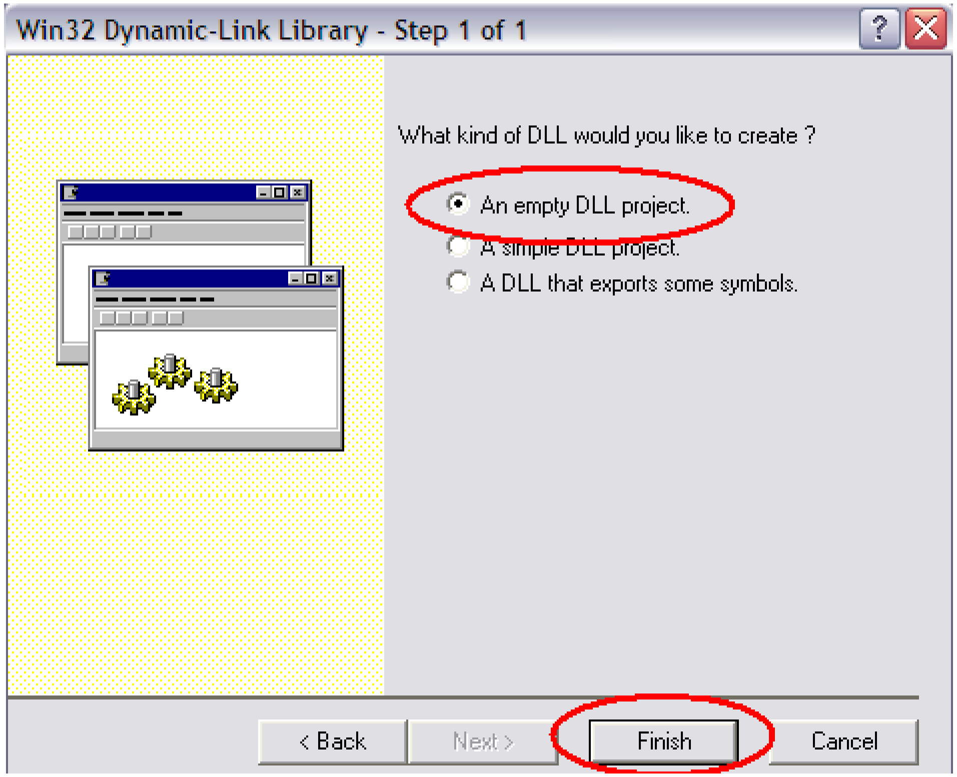
● Create new workspace

○ Add to current workspace

□ Dependency of:

Platforms: ☑ Win32

XYltoYp – Implementação MSVC / MEX



Win32 Dynamic-Link Library - Step 1 of 1

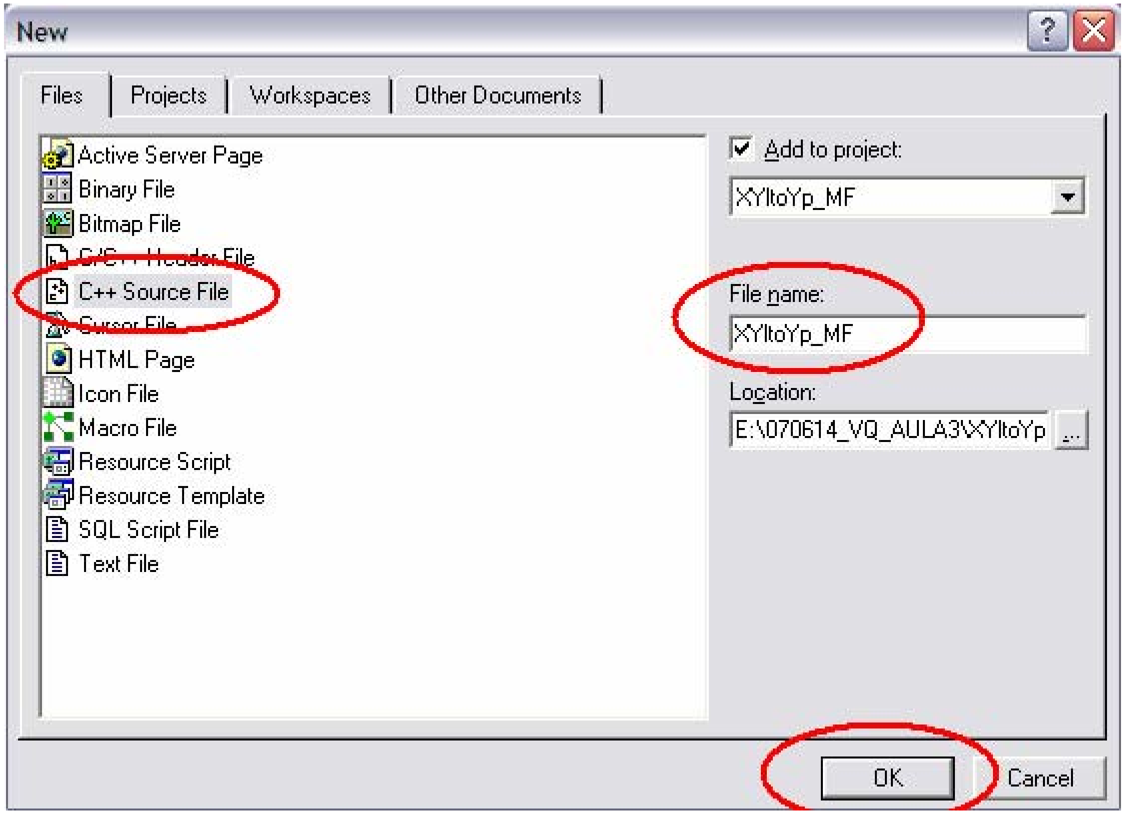
What kind of DLL would you like to create ?

● An empty DLL project.

○ A simple DLL project.

○ A DLL that exports some symbols.

XYltoYp – Implementação MSVC / MEX

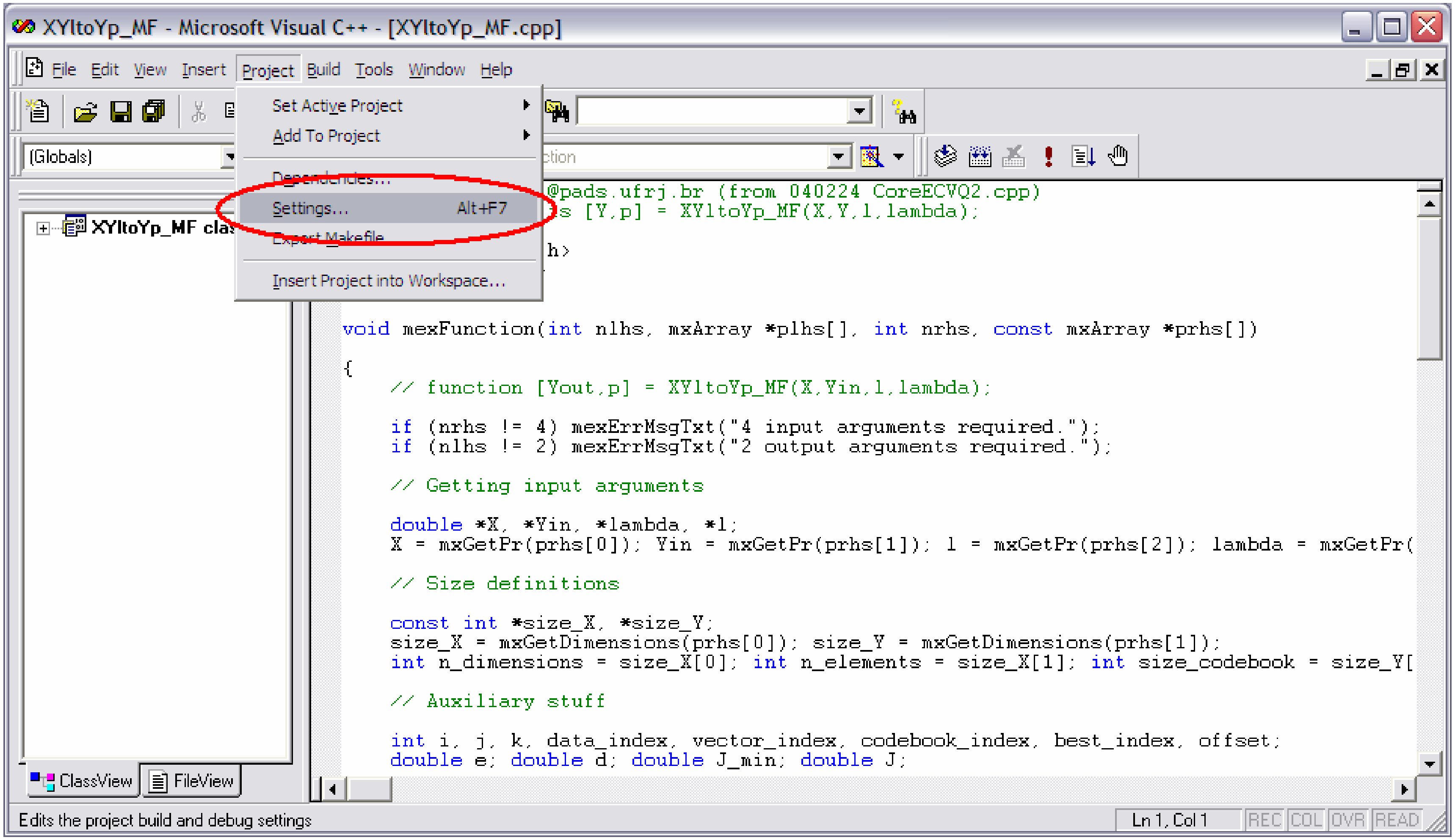


☑ Add to project: XYltoYp_MF

File name: XYltoYp_MF

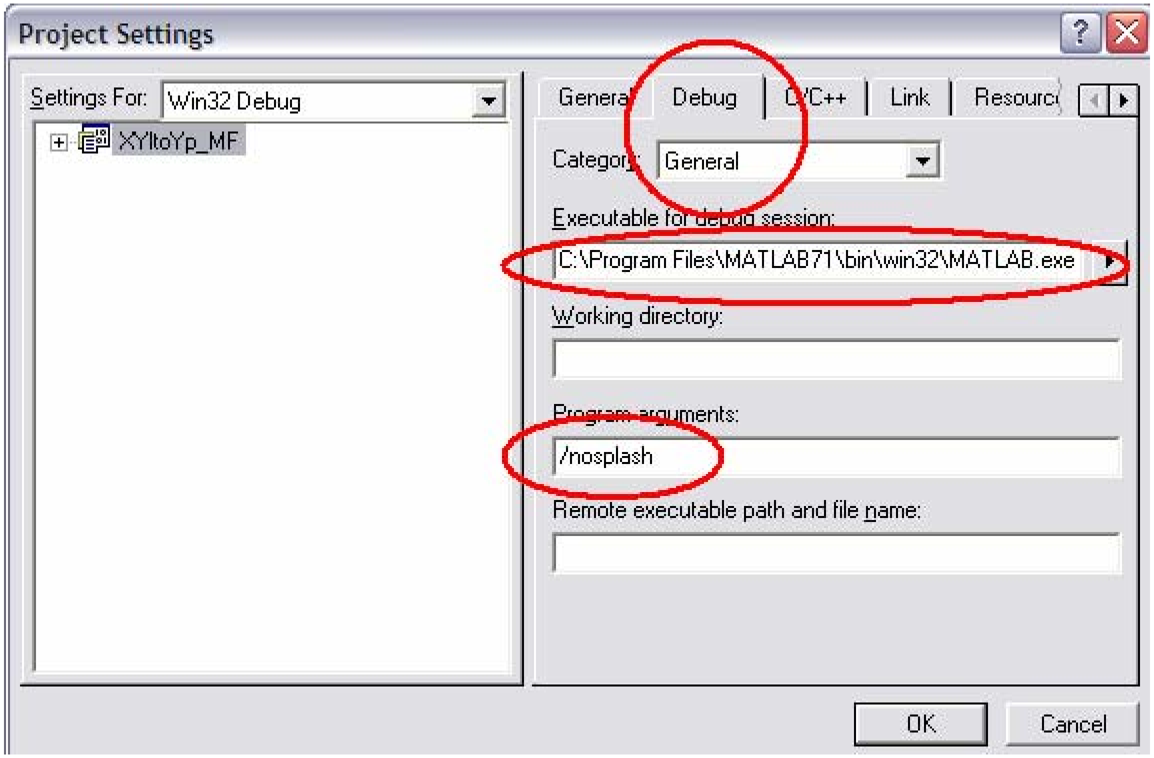
Location: E:\070614_VQ_AULA3\XYltoYp

XYltoYp – Implementação MSVC / MEX



```
@pads.ufrj.br (from 040224 CoreECVQ2.cpp)
s [Y,p] = XYltoYp_MF(X,Y,l,lambda);

h>


void mexFunction(int nlhs, mxArray *plhs[], int nrhs, const mxArray *prhs[])

{
    // function [Yout,p] = XYltoYp_MF(X,Yin,l,lambda);

    if (nrhs != 4) mexErrMsgTxt("4 input arguments required.");
    if (nlhs != 2) mexErrMsgTxt("2 output arguments required.");

    // Getting input arguments

    double *X, *Yin, *lambda, *l;
    X = mxGetPr(prhs[0]); Yin = mxGetPr(prhs[1]); l = mxGetPr(prhs[2]); lambda = mxGetPr(

    // Size definitions

    const int *size_X, *size_Y;
    size_X = mxGetDimensions(prhs[0]); size_Y = mxGetDimensions(prhs[1]);
    int n_dimensions = size_X[0]; int n_elements = size_X[1]; int size_codebook = size_Y[

    // Auxiliary stuff

    int i, j, k, data_index, vector_index, codebook_index, best_index, offset;
    double e; double d; double J_min; double J;
```

XYltoYp – Implementação MSVC / MEX



Settings For: Win32 Debug

Category: General

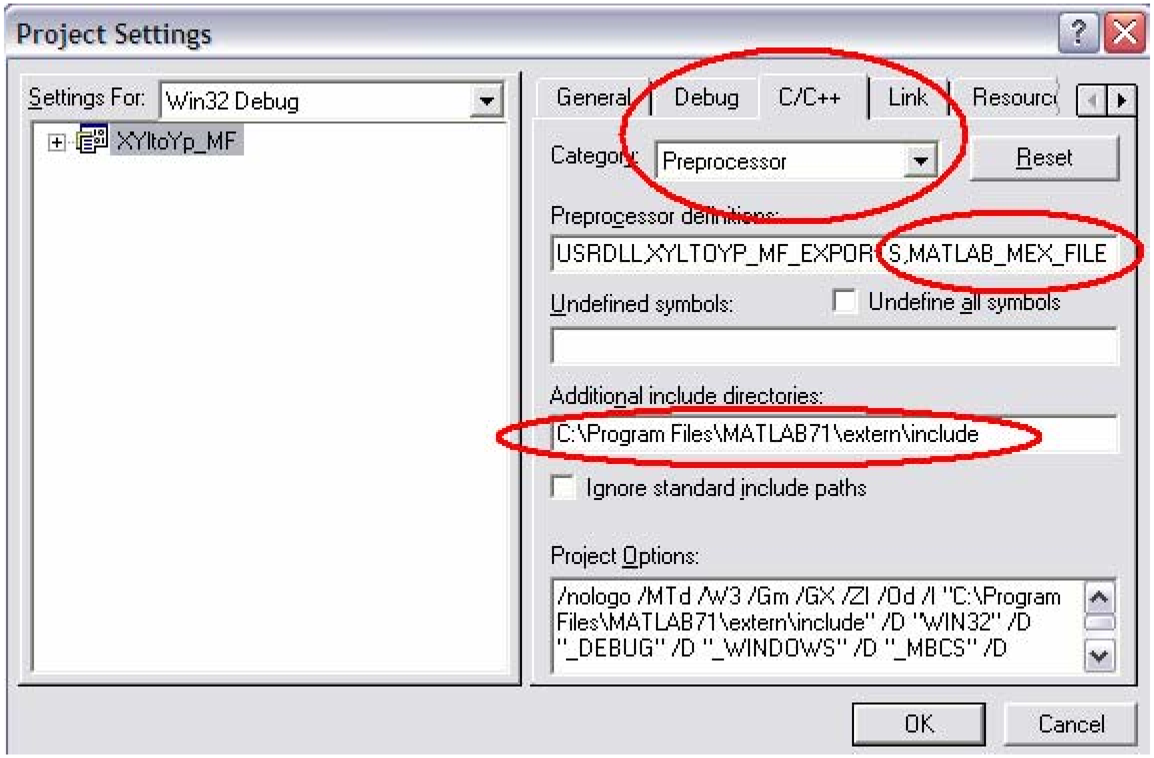
Executable for debug session: C:\Program Files\MATLAB71\bin\win32\MATLAB.exe

Working directory:

Program arguments: /nosplash

Remote executable path and file name:

XYltoYp – Implementação MSVC / MEX



Settings For: Win32 Debug

XYltoYp_MF

Category: Preprocessor

Preprocessor definitions:

USRDLL,XYLTOYP_MF_EXPORTS,MATLAB_MEX_FILE

Undefined symbols:

□ Undefine all symbols

Additional include directories:

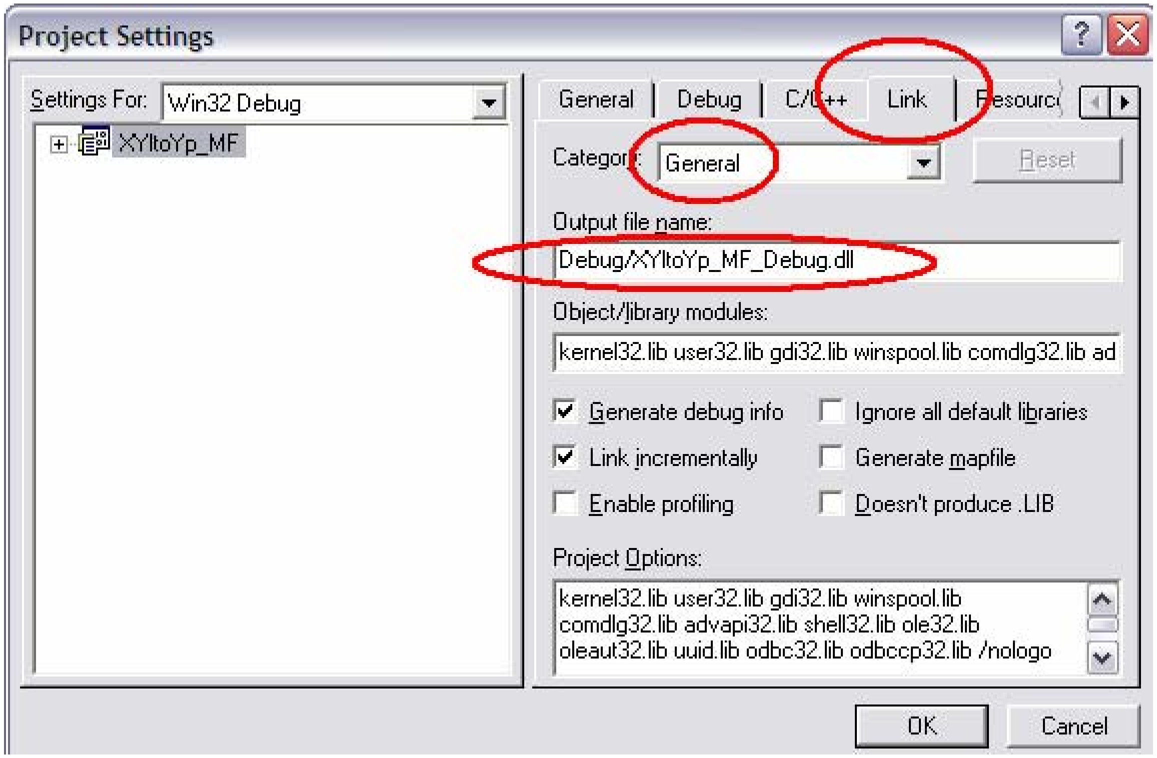
C:\Program Files\MATLAB71\extern\include

□ Ignore standard include paths

Project Options:

```
/nologo /MTd /W3 /Gm /GX /ZI /Od /I "C:\Program
Files\MATLAB71\extern\include" /D "WIN32" /D
"_DEBUG" /D "_WINDOWS" /D "_MBCS" /D
```

XYltoYp – Implementação MSVC / MEX



Settings For: Win32 Debug

XYltoYp_MF

Category: General

Output file name: Debug/XYltoYp_MF_Debug.dll

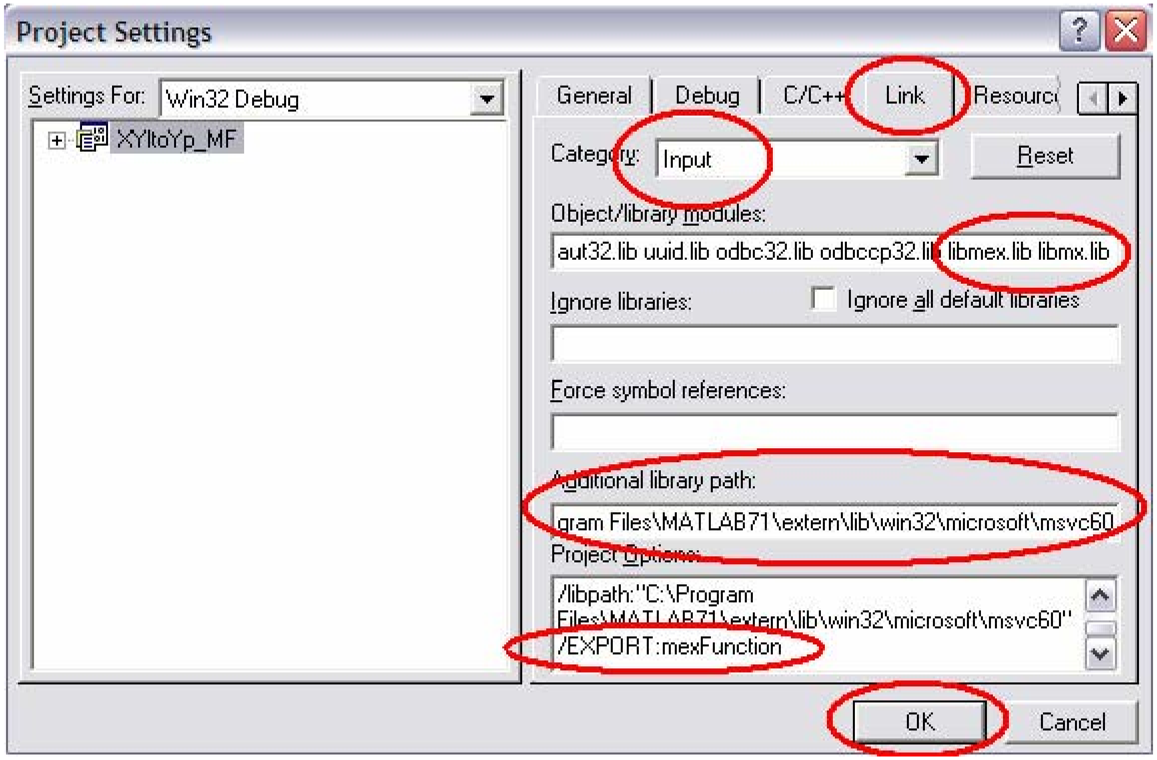
Object/library modules: kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib ad

☑ Generate debug info
☐ Ignore all default libraries
☑ Link incrementally
☐ Generate mapfile
☐ Enable profiling
☐ Doesn't produce .LIB

Project Options:

```
kernel32.lib user32.lib gdi32.lib winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib
oleaut32.lib uuid.lib odbc32.lib odbccp32.lib /nologo
```

XYltoYp – Implementação MSVC / MEX



Settings For: Win32 Debug

XYltoYp_MF

Category: Input

Object/library modules: aut32.lib uuid.lib odbc32.lib odbccp32.lib libmex.lib libmx.lib

Ignore libraries:

□ Ignore all default libraries

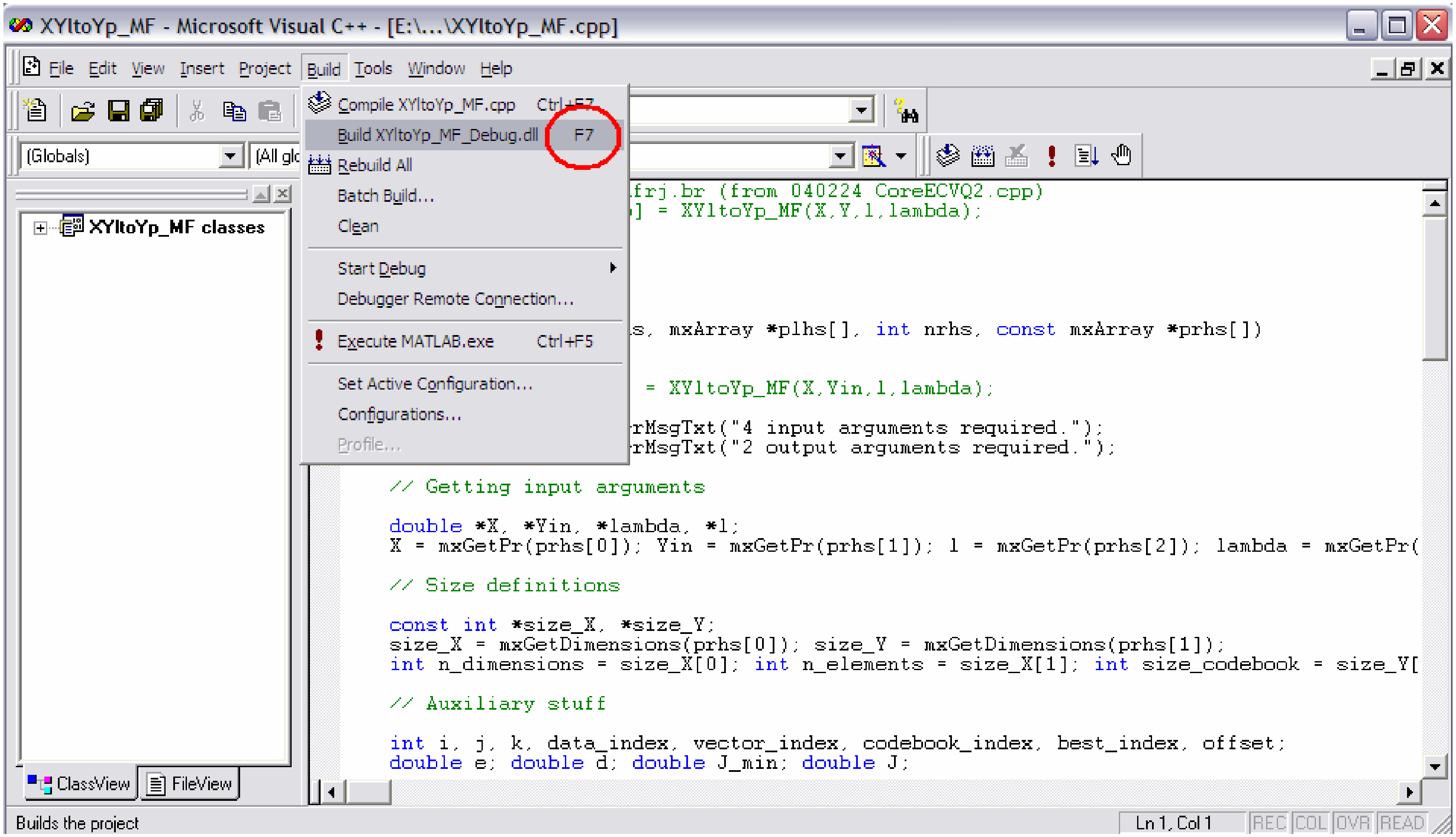
Force symbol references:

Additional library path: gram Files\MATLAB71\extern\lib\win32\microsoft\msvc60

Project Options:

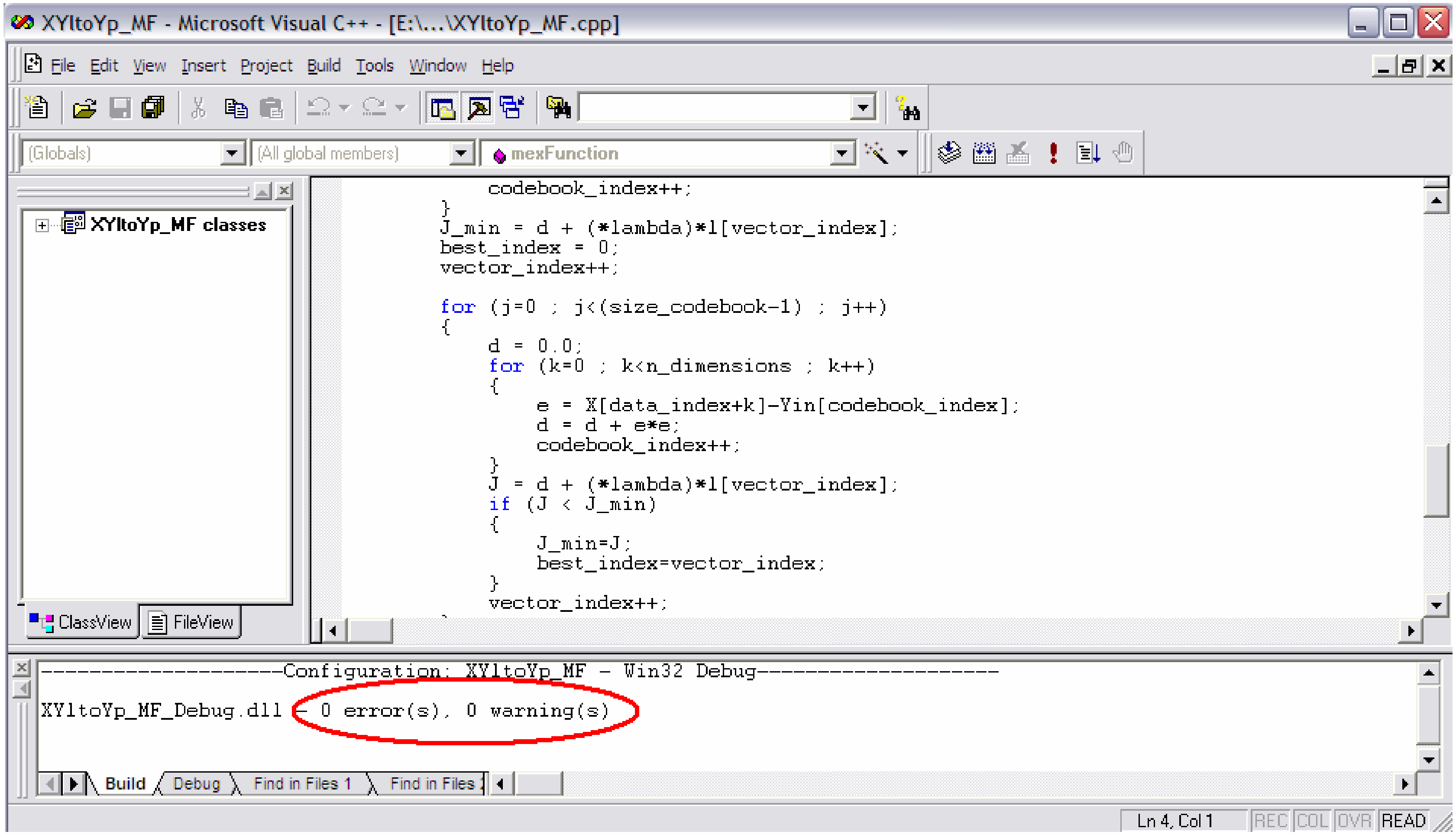
```
/libpath:"C:\Program
Files\MATLAB71\extern\lib\win32\microsoft\msvc60"
/EXPORT:mexFunction
```

XYltoYp – Implementação MSVC / MEX


XYltoYp_MF - Microsoft Visual C++ - [E:\...\XYltoYp_MF.cpp]
File Edit View Insert Project Build Tools Window Help
(Globals)
(All glo
XYltoYp_MF classes
Compile XYltoYp_MF.cpp Ctrl+F7
Build XYltoYp_MF_Debug.dll F7
Rebuild All
Batch Build...
Clean
Start Debug
Debugger Remote Connection...
Execute MATLAB.exe Ctrl+F5
Set Active Configuration...
Configurations...
Profile...
ClassView
FileView
Builds the project
Ln 1, Col 1
REC COL OVR READ

```
                                        .frj.br (from 040224 CoreECVQ2.cpp)
                                        ] = XYltoYp_MF(X,Y,l,lambda);


                                        s, mxArray *plhs[], int nrhs, const mxArray *prhs[])


                                         = XYltoYp_MF(X,Yin,l,lambda);

                                        rMsgTxt("4 input arguments required.");
                                        rMsgTxt("2 output arguments required.");

    // Getting input arguments

    double *X, *Yin, *lambda, *l;
    X = mxGetPr(prhs[0]); Yin = mxGetPr(prhs[1]); l = mxGetPr(prhs[2]); lambda = mxGetPr(

    // Size definitions

    const int *size_X, *size_Y;
    size_X = mxGetDimensions(prhs[0]); size_Y = mxGetDimensions(prhs[1]);
    int n_dimensions = size_X[0]; int n_elements = size_X[1]; int size_codebook = size_Y[

    // Auxiliary stuff

    int i, j, k, data_index, vector_index, codebook_index, best_index, offset;
    double e; double d; double J_min; double J;
```

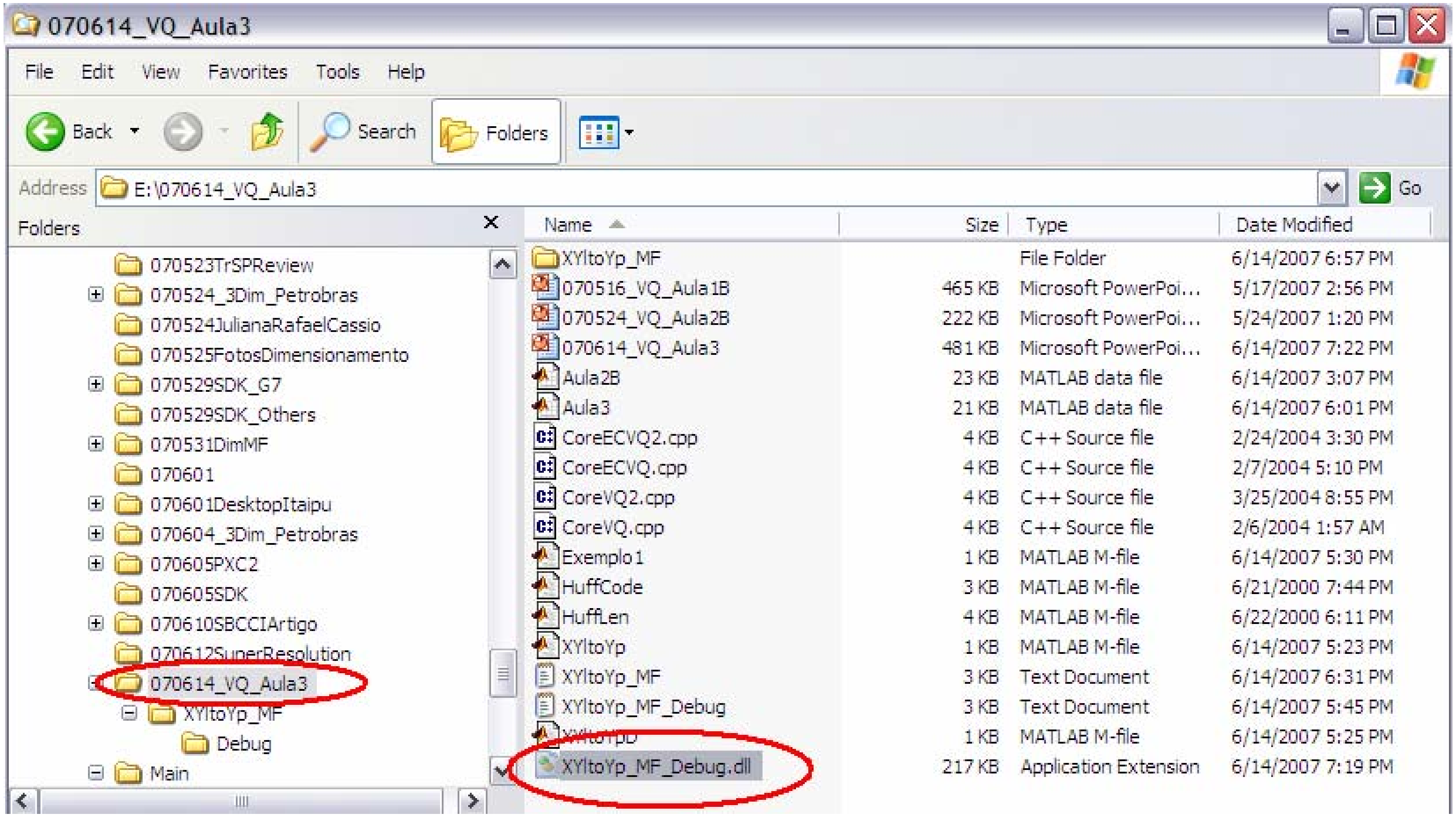
XYltoYp – Implementação MSVC / MEX



```
        codebook_index++;
    }
    J_min = d + (*lambda)*l[vector_index];
    best_index = 0;
    vector_index++;

    for (j=0 ; j<(size_codebook-1) ; j++)
    {
        d = 0.0;
        for (k=0 ; k<n_dimensions ; k++)
        {
            e = X[data_index+k]-Yin[codebook_index];
            d = d + e*e;
            codebook_index++;
        }
        J = d + (*lambda)*l[vector_index];
        if (J < J_min)
        {
            J_min=J;
            best_index=vector_index;
        }
        vector_index++;
```

```
--------------------Configuration: XYltoYp_MF - Win32 Debug--------------------
XYltoYp_MF_Debug.dll - 0 error(s), 0 warning(s)
```

XYltoYp – Implementação MSVC / MEX



E:\070614_VQ_Aula3

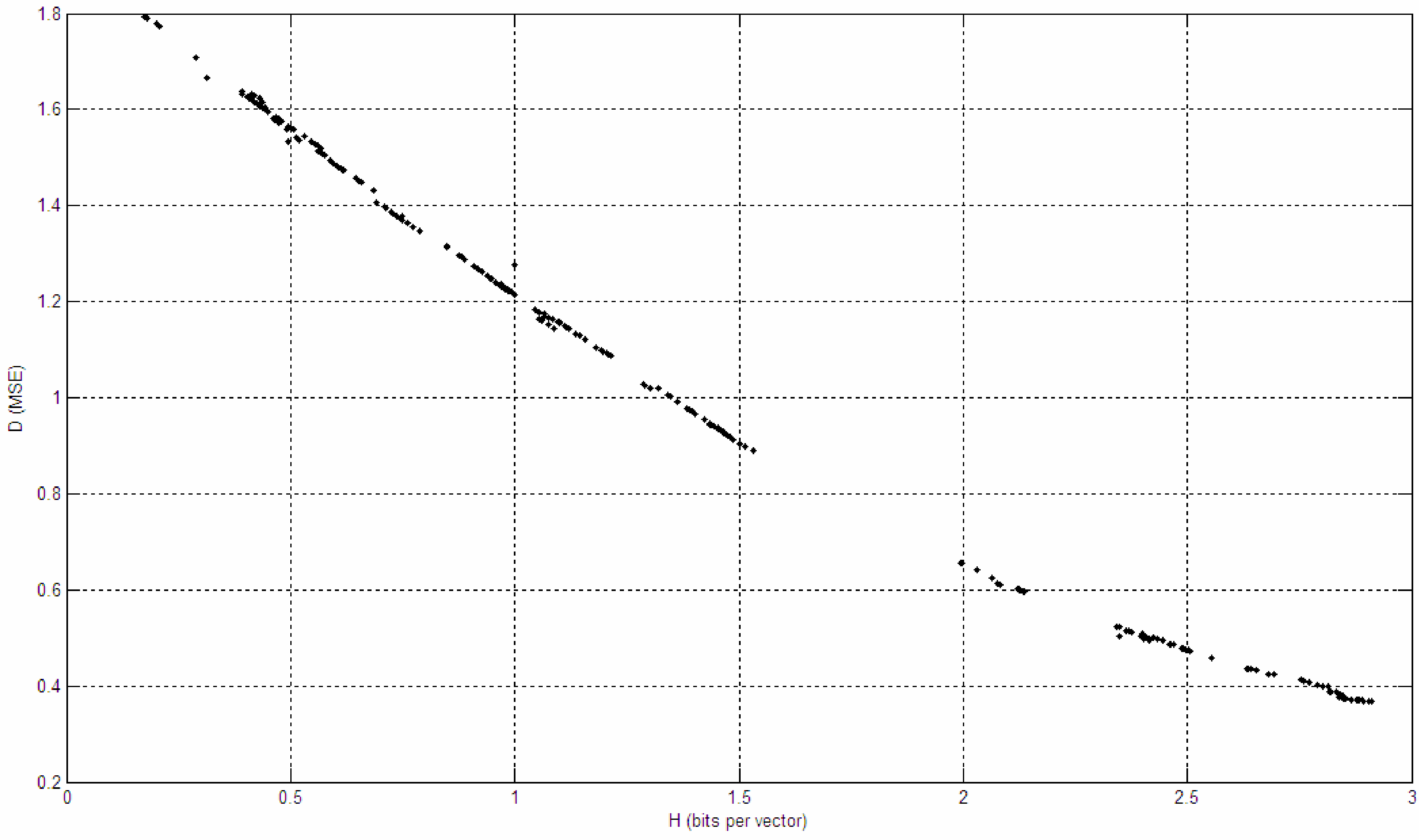
- 070523TrSPReview
- 070524_3Dim_Petrobras
- 070524JulianaRafaelCassio
- 070525FotosDimensionamento
- 070529SDK_G7
- 070529SDK_Others
- 070531DimMF
- 070601
- 070601DesktopItaipu
- 070604_3Dim_Petrobras
- 070605PXC2
- 070605SDK
- 070610SBCCIArtigo
- 070612SuperResolution
- 070614_VQ_Aula3
  - XYltoYp_MF
    - Debug
- Main

Name	Size	Type	Date Modified
XYltoYp_MF		File Folder	6/14/2007 6:57 PM
070516_VQ_Aula1B	465 KB	Microsoft PowerPoi...	5/17/2007 2:56 PM
070524_VQ_Aula2B	222 KB	Microsoft PowerPoi...	5/24/2007 1:20 PM
070614_VQ_Aula3	481 KB	Microsoft PowerPoi...	6/14/2007 7:22 PM
Aula2B	23 KB	MATLAB data file	6/14/2007 3:07 PM
Aula3	21 KB	MATLAB data file	6/14/2007 6:01 PM
CoreECVQ2.cpp	4 KB	C++ Source file	2/24/2004 3:30 PM
CoreECVQ.cpp	4 KB	C++ Source file	2/7/2004 5:10 PM
CoreVQ2.cpp	4 KB	C++ Source file	3/25/2004 8:55 PM
CoreVQ.cpp	4 KB	C++ Source file	2/6/2004 1:57 AM
Exemplo1	1 KB	MATLAB M-file	6/14/2007 5:30 PM
HuffCode	3 KB	MATLAB M-file	6/21/2000 7:44 PM
HuffLen	4 KB	MATLAB M-file	6/22/2000 6:11 PM
XYltoYp	1 KB	MATLAB M-file	6/14/2007 5:23 PM
XYltoYp_MF	3 KB	Text Document	6/14/2007 6:31 PM
XYltoYp_MF_Debug	3 KB	Text Document	6/14/2007 5:45 PM
XYltoYpD	1 KB	MATLAB M-file	6/14/2007 5:25 PM
XYltoYp_MF_Debug.dll	217 KB	Application Extension	6/14/2007 7:19 PM

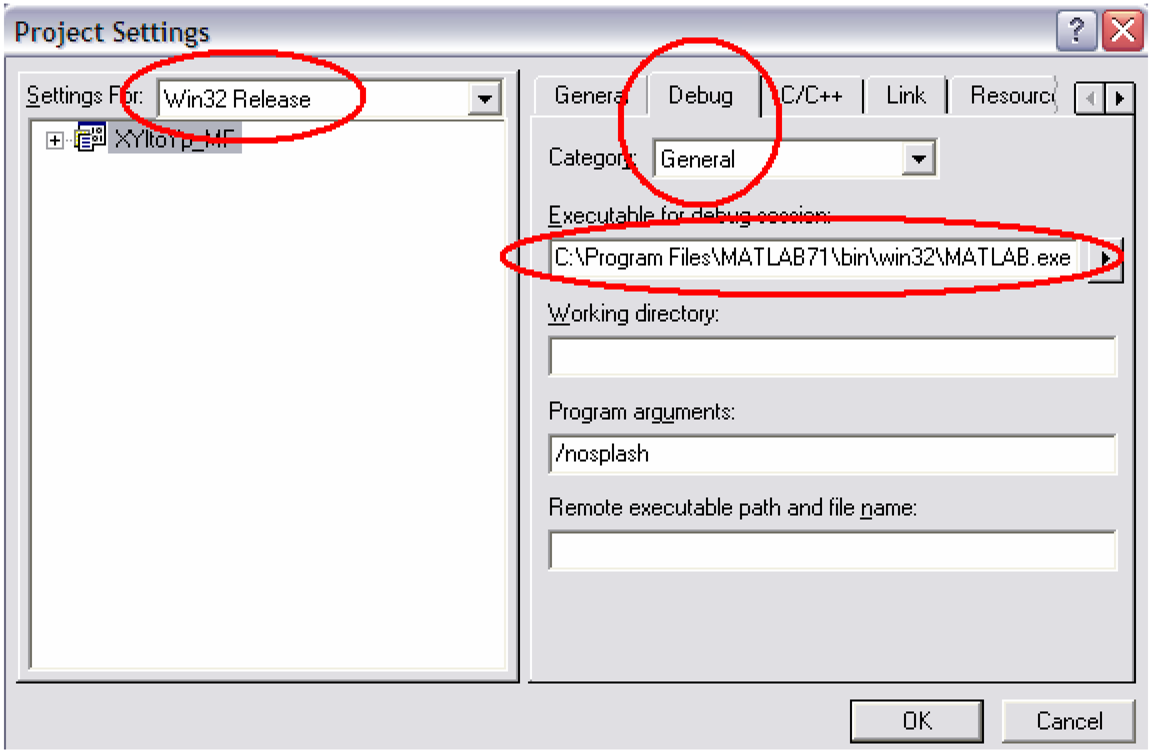
Programa Básico Modificado

```
clear all; close all; S = 0.01; BKJ = [];

for s = 1:400,

    lambda = S*(s-1);
    randn('state',0); rand('state',0); M = 2; N = 800; K = 8; e = 0.5;
    X = randn(M,N);
    Y = 0.5*randn(M,K);
    l = log2(K)*ones(1,size(Y,2));
    F = 200; BK = zeros(F,4);

    for i=1:F-1,
        [Y,p] = XYltoYp_MF_Debug(X,Y,l,lambda);
        Y = Y(:,find(p~=0));
        p = p(find(p~=0));
        p = p/sum(p);
        % Codeword Length Update
        l = HuffLen(p);
    end;

    [Y,p,D] = XYltoYpD(X,Y,l,lambda);
    p = p(find(p~=0)); p = p/sum(p);
    H = -sum(p.*log2(p));

    BKJ = [BKJ ; [lambda D H D+lambda*H size(Y,2)]]; % [s lambda D H D+lambda*H size(Y,2)]

end;

plot(BKJ(:,3),BKJ(:,2),'k.'); grid on; xlabel('H (bits per vector)'); ylabel('D (MSE)');

save Aula3B;
```

Programa Básico Modificado



Modo Release



Settings For: Win32 Release

XYItoYp_MF

Category: General

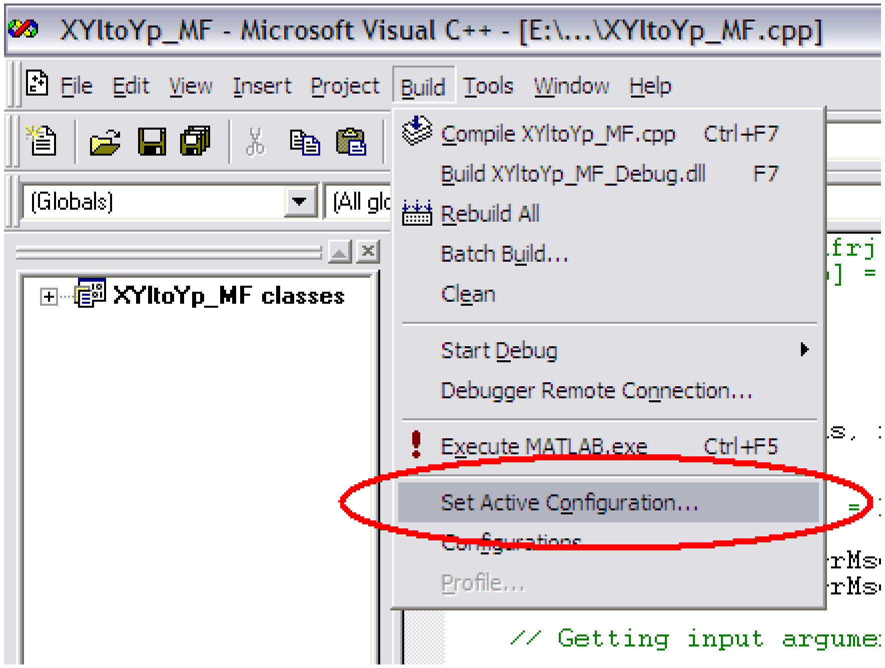
Executable for debug session:
C:\Program Files\MATLAB71\bin\win32\MATLAB.exe

Working directory:

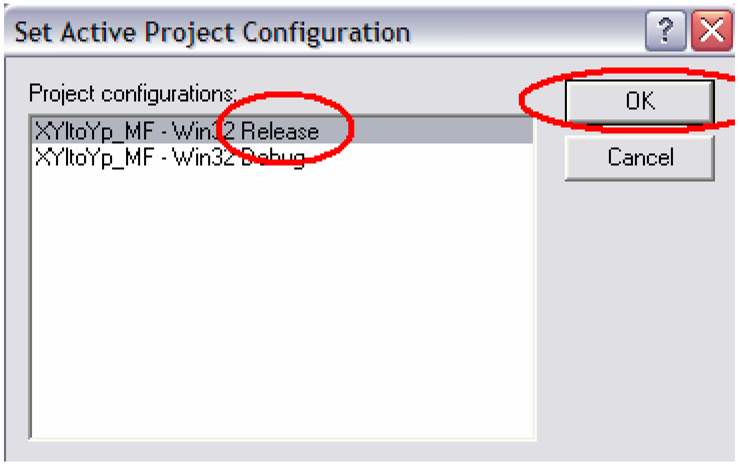
Program arguments:
/nosplash

Remote executable path and file name:

Modo Release


XYltoYp_MF - Microsoft Visual C++ - [E:\...\XYltoYp_MF.cpp]
File Edit View Insert Project Build Tools Window Help
(Globals)
(All glo
XYltoYp_MF classes
Compile XYltoYp_MF.cpp Ctrl+F7
Build XYltoYp_MF_Debug.dll F7
Rebuild All
Batch Build...
Clean
Start Debug
Debugger Remote Connection...
Execute MATLAB.exe Ctrl+F5
Set Active Configuration...
Configurations...
Profile...

```
// Getting input argume
```



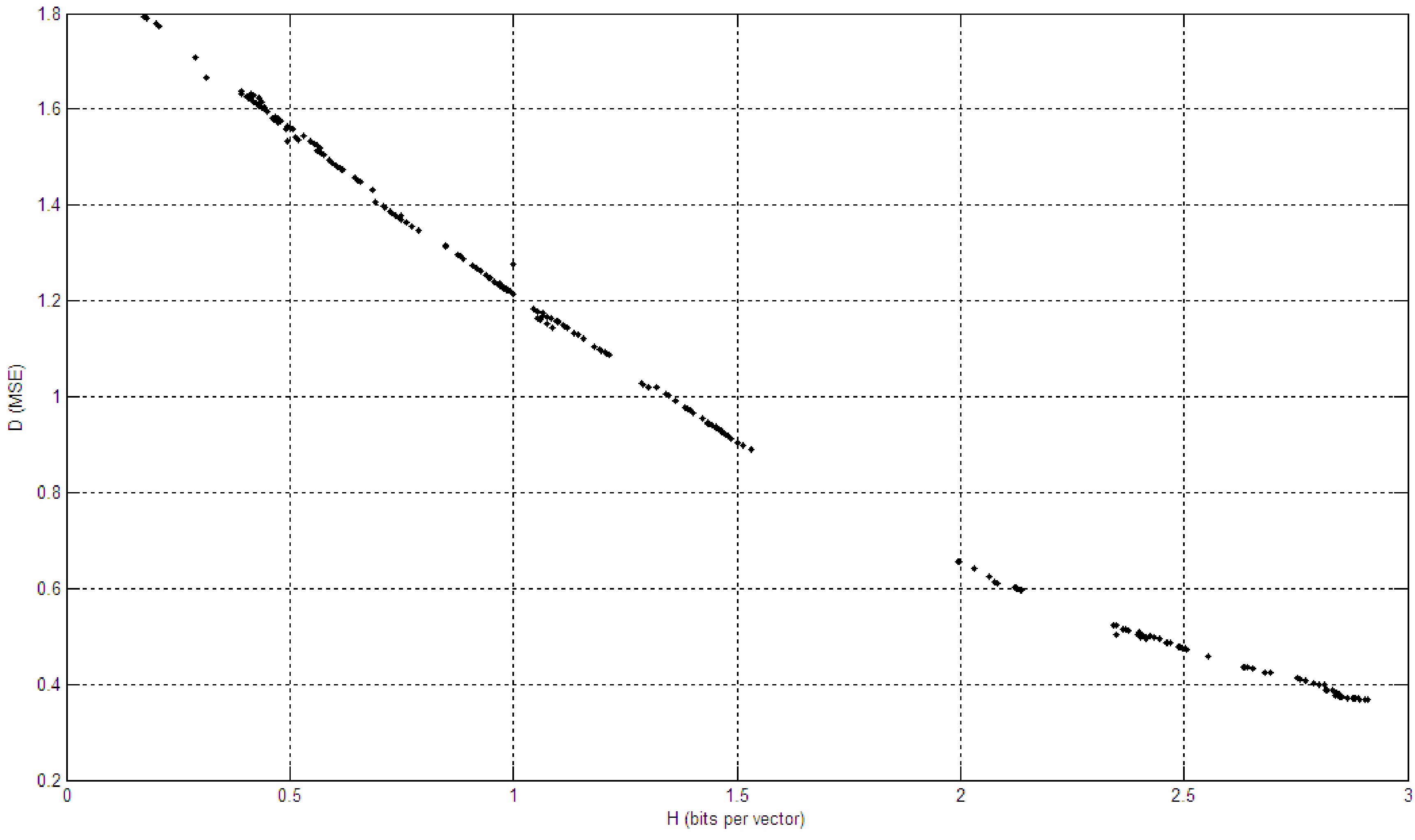
Project configurations:

XYltoYp_MF - Win32 Release
XYltoYp_MF - Win32 Debug

Programa Básico Modificado

```
clear all; close all; S = 0.01; BKJ = [];

for s = 1:400,

    lambda = S*(s-1);
    randn('state',0); rand('state',0); M = 2; N = 800; K = 8; e = 0.5;
    X = randn(M,N);
    Y = 0.5*randn(M,K);
    l = log2(K)*ones(1,size(Y,2));
    F = 200; BK = zeros(F,4);

    for i=1:F-1,
        [Y,p] = XYltoYp_MF(X,Y,l,lambda);
        Y = Y(:,find(p~=0));
        p = p(find(p~=0));
        p = p/sum(p);
        % Codeword Length Update
        l = HuffLen(p);
    end;

    [Y,p,D] = XYltoYpD(X,Y,l,lambda);
    p = p(find(p~=0)); p = p/sum(p);
    H = -sum(p.*log2(p));

    BKJ = [BKJ ; [lambda D H D+lambda*H size(Y,2)]]; [s lambda D H D+lambda*H size(Y,2)]

end;

plot(BKJ(:,3),BKJ(:,2),'k.'); grid on; xlabel('H (bits per vector)'); ylabel('D (MSE)');

save Aula3C;
```

Programa Básico Modificado



mex XYltoYp_MF_MATLAB.cpp;

```
mex -setup;
mex XYltoYp_MF_MATLAB.cpp;

clear all; close all; S = 0.01; BKJ = [];

for s = 1:400,

    lambda = S*(s-1);
    randn('state',0); rand('state',0); M = 2; N = 800; K = 8; e = 0.5;
    X = randn(M,N);
    Y = 0.5*randn(M,K);
    l = log2(K)*ones(1,size(Y,2));
    F = 200; BK = zeros(F,4);

    for i=1:F-1,
        [Y,p] = XYltoYp_MF_MATLAB(X,Y,l,lambda);
        Y = Y(:,find(p~=0));
        p = p(find(p~=0));
        p = p/sum(p);
        % Codeword Length Update
        l = HuffLen(p);
    end;

    [Y,p,D] = XYltoYpD(X,Y,l,lambda);
    p = p(find(p~=0)); p = p/sum(p);
    H = -sum(p.*log2(p));

    BKJ = [BKJ ; [lambda D H D+lambda*H size(Y,2)]]; % [s lambda D H D+lambda*H size(Y,2)]

end;

plot(BKJ(:,3),BKJ(:,2),'k.'); grid on; xlabel('H (bits per vector)'); ylabel('D (MSE)');

save Aula3D;
```