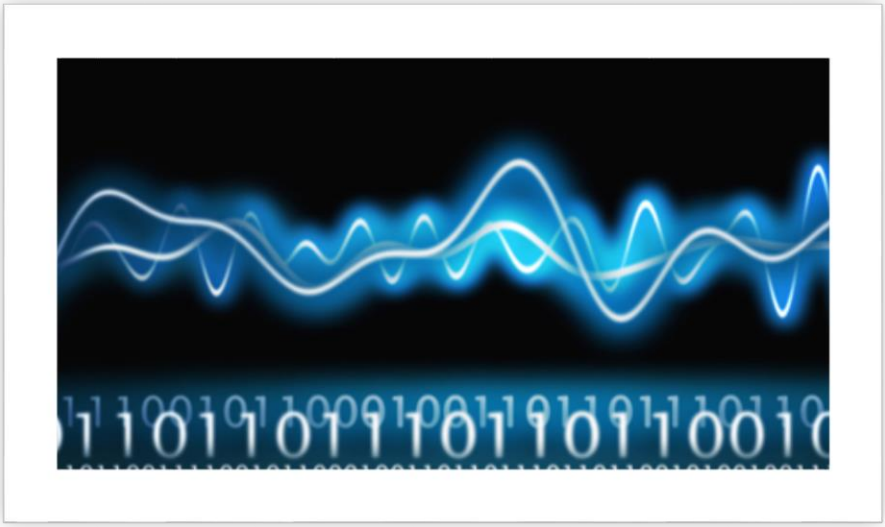
Universidade do Minho

Detecção de segmentos de fala

PROCESSOS ESTOCÁSTICOS



Ricardo Alves Teixeira – A62022
Processamento Digital de Sinal - Mestrado Integrado em Engenharia de Electrónica Industrial e Computadores

Guimarães, Julho de 2013

Universidade do Minho

Detecção de segmemtnos defala

Processos estocásticos

Ricardo Alves Teixeira – A62022

Processamento Digital de Sinal - Mestrado Integrado em Engenharia de Electrónica Industrial e Computadores lecionado pelo docente Carlos Lima

Guimarães, Julho de 2013

Conteúdo

Introdução

Foi proposto implementar um algoritmo para a deteção de intervalos de silêncio na fala.

Em comunicações estes intervalos são desprezados e não são enviados pois não contêm informação linguística.

Pretende-se usar o conhecido "Shewart Protocol" mas com *threshold* adaptativo às condições do ruído. Este mecanismo requer:

1. Estudo heurístico do melhor valor de *threshold* para cada relação sinal-ruído. Pretende-se que faça este estudo para SNR de 0 a 50 dB com intervalos de 10 dB. Este estudo requer síntese de ruído branco, soma ao sinal e deteção dos segmentos contendo apenas ruído usando vários valores do *threshold*, valores entre 0.2 e 5, que devem ser colocados numa tabela.
2. Cálculo automático da SNR e aplicação dos valores calculados no ponto 1).

Fundamentos teóricos

RELAÇÃO SINAL-RUÍDO (SNR)

A relação sinal-ruído consiste numa operação de comparação entre os níveis do sinal que espera captar e o ruído de fundo. Tal é definido como a razão entre a potência do sinal e a potência do ruído, expressa normalmente em dB. Uma razão superior a 1:1 (o que equivale a 0 dB) indica níveis de sinal superiores a níveis de ruído.

O cálculo do SNR pode também ser dado por,

$\mathrm{SNR} = \frac{\mu}{\sigma}$ em que μ é a média do sinal e ∂ é o desvio-padrão do ruído ou uma estimativa deste.

RUÍDO BRANCO

Em processamento de sinal, o ruído branco consiste num sinal aleatório com uma potência espectral constante. Tal implica que o sinal possui uma potência igual em qualquer banda de frequência.

Também se aplica este termo em sinais discretos dos quais as amostras são tomadas como uma sequência de variáveis aleatórios não-correladas com média nula e variância finita. Se cada amostra tem uma distribuição normal com média nula, o sinal é denominado de ruído branco gaussiano.

PROCESSOS ESTOCÁSTICOS

Enquanto que no caso dos processos determinísticos, sabemos à partida o conteúdo de um dado sinal, quer pela enumeração de valores não-nulos, quer através da representação de uma função x[n].

No entanto, para sinais em que não é possível saber do seu conteúdo previamente, como por exemplo, o que vai ser dito a seguir. Mesmo não sabendo previamente o que efetivamente foi falado no discurso, sabe-se à partida que se espera a captação de um discurso/fala, portanto, pode-se descrever os sinais estocásticos através de um modelo probabilístico. Para além disso, é possível efetuar processamento de sinal sobre sinais estocásticos através dos mesmos métodos usados em processos determinísticos.

Podemos considerar sinais estocásticos como sinais aleatórios, cujas amostras são independentes de todas as outras, pelo que sempre que repetimos as experiências e registamos o seu output, os valores das sucessivas amostras serão de um modo geral diferentes das amostras obtidas nas experiências anteriores. Por outras palavras, não é possível obter um padrão de como é que o sistema se comporta, se por exemplo determinarmos a FFT ou DFT de um dado sinal seja com quantos pontos forem.

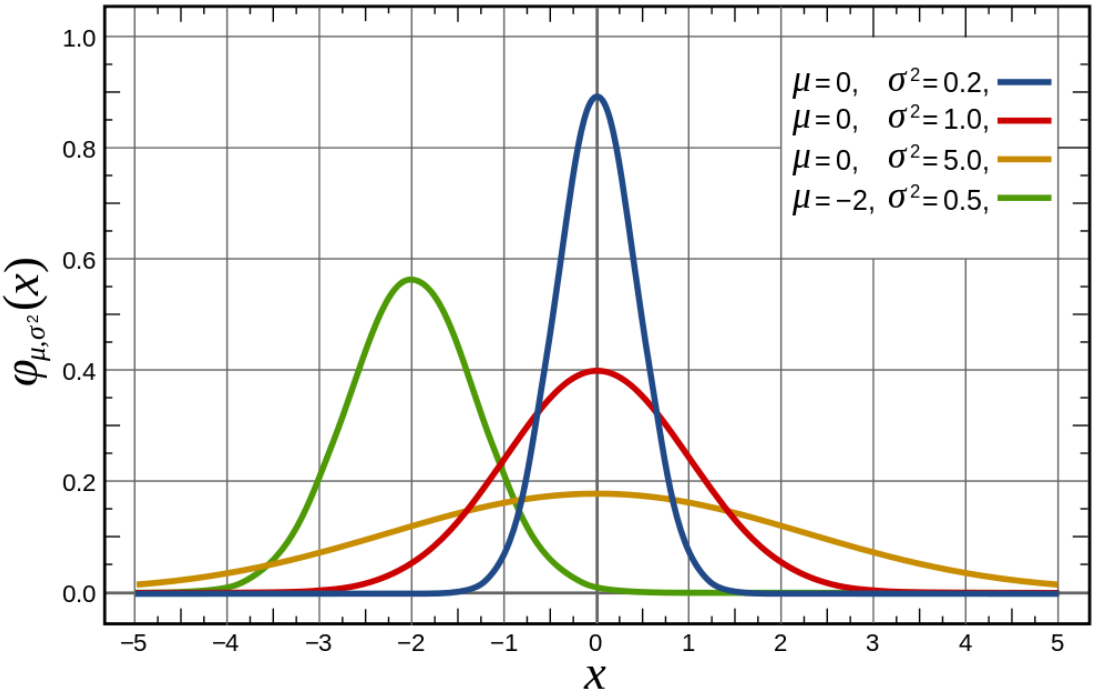
Quando estamos na presença de dados aleatórios, a estratégia passa por calcular “médias”.

Ao obtermos a distribuição normal de ruído branco do sinal, é possível saber os limites a partir dos quais um segmento do sinal é efetivamente ruído ou contém informação relevante (aplicando ao problema em estudo, considera-se a informação relevante como sendo a fala).

Na área das comunicações, recorre-se ao processos estocásticos para detetar intervalos de silêncio, em que nenhuma informação relevante está a ser passada pelo canal de comunicação, para suspender o envio de pacotes de dados de modo a não utilizar desnecessariamente os recursos de um sistema de comunicação.

DETERMINAÇÃO DO THRESHOLD

Obtido o modelo do ruído sob a forma de uma distribuição gaussiana e dado um valor médio do ruído, a atribuição de diferentes valores à variância tem influência sobre o valor de *threshold* a partir do qual se pode considerar uma amostra como não sendo parte do ruído.



SHEWHART PROTOCOL

Walter A. Shewhart inventou este protocolo quando trabalhava para a companhia *Bell Labs* nos anos 1920. Visto que os sistemas de telecomunicações tiveram de ser enterrados no subsolo, houve a necessidade de arranjar uma forma de melhorar a sua fiabilidade, reduzindo a ocorrência de avarias e de reparações.

A partir de teorias estatísticas puramente matemáticas, *Shewhart* chegou à conclusão de que dados amostrados de processos físicos resultam de um modo geral numa curva de distribuição normal (distribuição gaussiana)

O gráfico de controlo implementado por Shewhart consiste em:

- Pontos que representam a estatística (média e limites por exemplo) de medições qualitativas sob a forma de amostras tiradas de processos físicos em altura diferentes.
- A média da estatística referida no ponto anterior (média da médias e média dos limites) é calculada
- Uma linha central é desenhada com o valor da média da estatística
- O erro-padrão (desvio-padrão em relação à média) é também calculado usando todas as amostras.

- Limites superiores e inferiores de controlo que indicam o threshold ao qual o output do processo é considerado estatisticamente improvável são definidos como sendo 3 vezes o desvio-padrão da linha central.

Se o processo está sob controlo e se a sua distribuição é normal, 99,73% de todas as amostras terão valores dentro dos limites definidos. Qualquer ocorrência fora dos limites indica a necessidade de investigar por eventuais problemas que estejam a afetar o processo.

Procedimentos experimentais

Através do programa Matlab R2013a implementar-se-á o algoritmo de deteção de segmentos de silêncio ao longo de um discurso, bem como serão usadas as funcionalidades disponíveis para o cálculo dos valores de *threshold*, adição de ruído branco ao sinal de áudio captado e numa segunda fase será implementado o cálculo automático do SNR e deteção automática dos intervalos de silêncio e fala.

MÉTODOS EXPERIMENTAIS PARA O ESTUDO HEURÍSTICO DO MELHOR VALOR DE THRESHOLD PARA CADA RELAÇÃO SINAL-RUÍDO

Primeiramente foi guardado na variável áudio um pequeno discurso, sendo que durante os primeiros segundos são compostos de ausência de fala, de modo a de seguida obtermos um modelo do ruído presente no sinal áudio.

```
audio=wavrecord(15*8000,8000);
```

Novas variáveis foram de seguida criadas, contendo integralmente o discurso captado na variável áudio, só que com a adição de ruído branco em todas as amostras de modo a estudarmos o melhor valor de *threshold* para SNR de 0dB a 50dB, com intervalos de 10dB. De modo a obter ruído branco usou-se uma função incluída no Matlab, cujo protótipo é:

```
y = awgn(x,snr,'measured')
```

Na linha de comandos do Matlab, procedeu-se da seguinte forma

```
>> asnr0=awgn(audio,0,'measured');

>> asnr10=awgn(audio,10,'measured');

>> asnr20=awgn(audio,20,'measured');

>> asnr30=awgn(audio,30,'measured');

>> asnr40=awgn(audio,40,'measured');

>> asnr50=awgn(audio,50,'measured');
```

A função que se segue calcula a média e a variância de parte do sinal que corresponde ao ruído, percorrendo de seguida a variável que contém o sinal de áudio, passando os segmentos de amostras que são consideradas como fala para a variável *speechbuf* e os intervalos de silêncio para a variável *noisebuf*. Caso existam segmentos de fala na variável *noisebuf*, é necessário ajustar os parâmetros que a função recebe.

Segue-se uma descrição detalhada de tal função:

A função retorna os argumentos:

- Speechbuf: variável que contém apenas os segmentos de fala
- Noisebuf: variável que contém os segmentos de ruído
- Nmean: média das amostras referentes ao ruído
- Nvar: variância das amostras referentes ao ruído
- Ndev: desvio-padrão das amostras referentes ao ruído

- Snr: relação sinal ruído do sinal em análise

Os parâmetros que a função recebe são

- Audiocap: que contem o sinal de áudio gravado
- Noiselen: número de pontos desde o início do array audiocap que correspondem a ruído
- Windlen: tamanho da janela que analisará um conjunto de amostras
- Alfa: fração da janela que deve ter pontos a 1 de modo a que o conjunto de amostras seja considerado como um segmento de fala.

A variável *auxarr* é em termos de tamanho igual ao número de amostras que o sinal de áudio contém, podendo conter em cada índice zeros (se a amostra for de ruído) ou uns (se a amostra corresponder à fala)

```
function [speechbuf, noisebuf, nmean, nvar, ndev, snr] = sigproc
(audiocap, noiselen, windlen, step, alfa)
    auxarr = zeros(size(audiocap,1),1);
    nmean = mean(audiocap(1:noiselen,1));
    nvar = var(audiocap(1:noiselen,1));
    ndev = sqrt(nvar);
```

Calculadas as média (*nmean*), variância (*nvar*) e desvio-padrão (*ndev*), obtemos um modelo do ruído. No ciclo for que se suceder, se o absoluto valor da amostra se situar fora do intervalo de valores entre 0 e *nmean* + *ndev*, então estamos na presença de uma amostra que pode constituir parte de um discurso.

```
    for i=1:length(audiocap)
        if abs(audiocap(i,1)-nmean) > ndev
            auxarr(i,1) = 1;
        else
            auxarr(i,1) = 0;
        end
    end
```

Para evitar a falsos positivos e as pausas durante um discurso, usa-se o método de análise à janela. Este método consiste em dividir a totalidade do sinal áudio num conjunto de *windlen* amostras. Se o número de uns nessa janela for superior a alfa vezes o tamanho da janela, então podemos considerar que o segmento analisado corresponde à fala, pelo que se adiciona à variável *speechbuf*.

As variáveis *begw* e *endw* correspondem aos limites inferiores e superiores da janela utilizada para analisar os conjuntos de amostras. Finda a análise de um segmento de amostras, a variável *begw* toma o valor da variável *endw*, sendo somada ao valor da variável *endw* o tamanho da janela *windlen*.

```
    begw = 1;
    endw = windlen;
    processing = 1;
    noisebuf=0;
    speechbuf=0;
    final = 0;
    while processing
           if final == 1
            processing = 0;
        end
        tempacap = auxarr(begw:endw,1);
        if sum(tempacap) > alfa*(endw-begw)
```

```
            speechbuf = cat(1,speechbuf, audiocap(begw:endw,1));
        else
            noisebuf= cat(1,noisebuf, audiocap(begw:endw,1));
        end
```

As seguintes condições *if* foram adicionadas ao código com o intuito de as variáveis *begw* e *endw* não tomarem valores superiores ao tamanho do *array* que está a ser analisado.

```
        if begw+windlen < length(audiocap)
            begw=endw;
        end

        if endw+windlen < length(audiocap)
            endw=endw+windlen;
```

Terminada a análise, a variável final toma o valor de um, de modo a sair do ciclo *while* e da função.

```
        else
            endw=length(audiocap);
            final=1;
        end


    end
    snr = 20*log(mean(speechbuf)/sqrt(var(noisebuf)));
end
```

Descrita a função, esta será invocada da seguinte maneira:

Criadas as variáveis com o sinal de áudio ao qual é adicionado ruído branco para um SNR de 0dB a 50dB em intervalos de 10dB, armazenam-se os valores das médias de ruído de cada sinal em stats(i,1), variância em stats(i,2) , desvio-padrão em stats(i,3) e o SNR stats(i,4).

```
asnr0=awgn(audio,0,'measured');
asnr10=awgn(audio,10,'measured');
asnr20=awgn(audio,20,'measured');
asnr30=awgn(audio,30,'measured');
asnr40=awgn(audio,40,'measured');
asnr50=awgn(audio,50,'measured');
i=1;
[speechbuf, noisebuf, stats(i,1), stats(i,2),stats(i,3),stats(i,4)] =
sigproc (asnr50, 8000, 130, 0.6);
i=i+1;
[speechbuf, noisebuf, stats(i,1), stats(i,2),stats(i,3),stats(i,4)] =
sigproc (asnr40, 8000, 130, 0.6);
i=i+1;
[speechbuf, noisebuf, stats(i,1), stats(i,2),stats(i,3),stats(i,4)] =
sigproc (asnr30, 8000, 130, 0.6);
i=i+1;
[speechbuf, noisebuf, stats(i,1), stats(i,2),stats(i,3),stats(i,4)] =
sigproc (asnr20, 8000, 130, 0.6);
i=i+1;
[speechbuf, noisebuf, stats(i,1), stats(i,2),stats(i,3),stats(i,4)] =
sigproc (asnr10, 8000, 130, 0.6);
i=i+1;
[speechbuf, noisebuf, stats(i,1), stats(i,2),stats(i,3),stats(i,4)] =
sigproc (asnr0, 8000, 130, 0.6);
```

Por último calcula-se a média das médias obtidas e média do desvio-padrão

```
globmean=mean(stats(1:5,1));
globdev=mean(stats(1:5,3));
```

CÁLCULO AUTOMÁTICO DO SNR E APLICAÇÃO DOS VALORES OBTIDOS

Calculadas as médias do desvios-padrão e médias obtidas para os sinais com diferente níveis de ruído, há a necessidade de adaptar a função *sigproc*, de modo a que se dispense o cálculo da média e variância do ruído de um determinado sinal de áudio, sendo que agora as média e desvio-padrão do modelo gaussiano de ruído obtidos serão passados como argumento à função *autosigproc*

```
function [speechbuf, noisebuf, snr] = autosigproc (audiocap,
globmean, globdev, windlen, alfa)
    auxarr = zeros(size(audiocap,1),1);
    nmean = globmean;
    ndev = globdev;

    for i=1:length(audiocap)
        if abs(audiocap(i,1)-nmean) > ndev
            auxarr(i,1) = 1;
        else
            auxarr(i,1) = 0;
        end
    end

    begw = 1;
    endw = windlen;
    processing = 1;
    noisebuf=0;
    speechbuf=0;
    final = 0;
    while processing
           if final == 1
            processing = 0;
        end
        tempacap = auxarr(begw:endw,1);
        if sum(tempacap) > alfa*(endw-begw)
            speechbuf = cat(1,speechbuf, audiocap(begw:endw,1));
        else
            noisebuf= cat(1,noisebuf, audiocap(begw:endw,1));
        end
        if begw+windlen < length(audiocap)
            begw=endw;
        end

        if endw+windlen < length(audiocap)
            endw=endw+windlen;
        else
            endw=length(audiocap);
            final=1;
        end

    end
  snr = 20*log(mean(speechbuf)/sqrt(var(noisebuf)));
end
```

Resultados obtidos

MÉTODOS EXPERIMENTAIS PARA O ESTUDO HEURÍSTICO DO MELHOR VALOR DE THRESHOLD PARA CADA RELAÇÃO SINAL-RUÍDO

Dispostos numa tabela encontram-se os valores de média e desvio-padrão da parte de ruído do sinal que foram obtidos para um *additive white gaussian noise* tal que SNR se encontra entre 0 dB e 50 dB

SNR (dB)	Média	Desvio-padrão
50	-0,0110312108357602	0,0220946696719108
40	-0,0110240912040719	0,0220908542888547
30	-0,0110266117058621	0,0221245482379478
20	-0,0110088139132300	0,0224166258650203
10	-0,0109178296623996	0,0254936058219257
0	-0,0104975358919722	0,0453124378642787

Média das médias: -0,0110017114642648

Média dos desvios-padrão: 0,0228440607771319

CÁLCULO AUTOMÁTICO DO SNR E APLICAÇÃO DOS VALORES OBTIDOS

Para detetar os segmentos de fala e de silêncio, usando a função *autosigproc*, passando como argumentos a média das médias e a média dos desvios-padrão, houve sucesso na realização da separação dos segmentos de silêncio e de fala, quer com os sinais de áudio usados ao longo do trabalho prático, quer com sinais de áudio gravados posteriormente.

Conclusões

Mesmo tendo havido sucesso em completar os objetivos do trabalho, os valores de *threshold* propostos no enunciado (0,2 a 5) não serviram para a resolução do problema tratado neste trabalho prático. Isto é devido ao facto de durante os testes, as amostras serem todas consideradas como ruído, visto que a médias e a variância do ruído analisado são muito inferiores ao valor mínimo de *threshold* proposto.

Uma má interpretação sobre o conceito de *threshold*, aplicado a este contexto pode também ter ocorrido, por falta de clarificação no enunciado em conjunto com a pouca informação existente nas fontes bibliográficas.

De qualquer maneira, tal como referido anteriormente os objetos podem ser considerados como cumpridos, visto que através do uso da função *autosigproc* a deteção dos intervalos de silêncio é levada a cabo, sendo eventualmente necessário ajustar apenas os valores da janela e o parâmetro alfa, se as circunstâncias assim o exigirem.

Por último, este trabalho permitiu aprofundar alguns dos conhecimentos sobre processos estocásticos, que são importantes em processamento de sinais do mundo real.

Bibliografia

Control chart. (s.d.). Obtido de Wikipedia: https://en.wikipedia.org/wiki/Control_chart

Normal distribution. (s.d.). Obtido de Wikipedia: https://en.wikipedia.org/wiki/Normal_distribution

Signal-to-noise ratio. (s.d.). Obtido de Wikipedia: http://en.wikipedia.org/wiki/Signal-to-noise_ratio

Stochastic signal processing. (s.d.). Obtido de Cousera: https://class.coursera.org/dsp-001/lecture/109

White noise. (s.d.). Obtido de Wikipedia: http://en.wikipedia.org/wiki/White_noise

Anexo

FUNÇÃO SIGPROC

```
function [speechbuf, noisebuf, nmean, nvar, ndev, snr] = sigproc
(audiocap, noiselen, windlen, alfa)
    auxarr = zeros(size(audiocap,1),1);
    nmean = mean(audiocap(1:noiselen));
    nvar = var(audiocap(1:noiselen));
    ndev = sqrt(nvar);

    for i=1:length(audiocap)
        if abs(audiocap(i,1)-nmean) > ndev
            auxarr(i,1) = 1;
        else
            auxarr(i,1) = 0;
        end
    end

    begw = 1;
    endw = windlen;
    processing = 1;
    noisebuf=0;
    speechbuf=0;
    final = 0;
    while processing
           if final == 1
            processing = 0;
        end
        tempacap = auxarr(begw:endw,1);
        if sum(tempacap) > alfa*(endw-begw)
            speechbuf = cat(1,speechbuf, audiocap(begw:endw,1));
        else
            noisebuf= cat(1,noisebuf, audiocap(begw:endw,1));
        end
        if begw+windlen < length(audiocap)
            begw=endw;
        end

        if endw+windlen < length(audiocap)
            endw=endw+windlen;
        else
            endw=length(audiocap);
            final=1;
        end

    end
  snr = 20*log(mean(speechbuf)/sqrt(var(noisebuf)));
end
```

FUNÇÃO AUTOSIGPROC

```
function [speechbuf, noisebuf, snr] = autosigproc (audiocap,
globmean, globdev, windlen, alfa)
    auxarr = zeros(size(audiocap,1),1);
    nmean = globmean;
    ndev = globdev;

    for i=1:length(audiocap)
        if abs(audiocap(i,1)-nmean) > ndev
            auxarr(i,1) = 1;
        else
            auxarr(i,1) = 0;
        end
    end

    begw = 1;
    endw = windlen;
    processing = 1;
    noisebuf=0;
    speechbuf=0;
    final = 0;
    while processing
           if final == 1
            processing = 0;
        end
        tempacap = auxarr(begw:endw,1);
        if sum(tempacap) > alfa*(endw-begw)
            speechbuf = cat(1,speechbuf, audiocap(begw:endw,1));
        else
            noisebuf= cat(1,noisebuf, audiocap(begw:endw,1));
        end
        if begw+windlen < length(audiocap)
            begw=endw;
        end

        if endw+windlen < length(audiocap)
            endw=endw+windlen;
        else
            endw=length(audiocap);
            final=1;
        end

    end
  snr = 20*log(mean(speechbuf)/sqrt(var(noisebuf)));
end
```

COMANDO PARA CÁCULOS AUXILIARES

```
asnr0=awgn(audio,0,'measured');
asnr10=awgn(audio,10,'measured');
asnr20=awgn(audio,20,'measured');
asnr30=awgn(audio,30,'measured');
asnr40=awgn(audio,40,'measured');
asnr50=awgn(audio,50,'measured');
i=1;
```

```
[speechbuf, noisebuf, stats(i,1), stats(i,2),stats(i,3),stats(i,4)] =
sigproc (asnr50, 8000, 130, 0.6);
i=i+1;
[speechbuf, noisebuf, stats(i,1), stats(i,2),stats(i,3),stats(i,4)] =
sigproc (asnr40, 8000, 130, 0.6);
i=i+1;
[speechbuf, noisebuf, stats(i,1), stats(i,2),stats(i,3),stats(i,4)] =
sigproc (asnr30, 8000, 130, 0.6);
i=i+1;
[speechbuf, noisebuf, stats(i,1), stats(i,2),stats(i,3),stats(i,4)] =
sigproc (asnr20, 8000, 130, 0.6);
i=i+1;
[speechbuf, noisebuf, stats(i,1), stats(i,2),stats(i,3),stats(i,4)] =
sigproc (asnr10, 8000, 130, 0.6);
i=i+1;
[speechbuf, noisebuf, stats(i,1), stats(i,2),stats(i,3),stats(i,4)] =
sigproc (asnr0, 8000, 130, 0.6);
globmean=mean(stats(1:5,1));
globdev=mean(stats(1:5,3));
```