Trabalho 2

Tiago Loureiro Chaves (187690)

MC920A - Introdução ao Processamento de Imagem Digital - 2s2019

Resumo

Este projeto teve como objetivo avaliar diferentes métodos de limiarização de imagens monocromáticas (*thresholing*) para o mapeando de seus pixels em preto (objeto) e branco (fundo), resultando em imagens binárias.

Comparou-se a forma clássica de limiarização global, onde um único *threshold* é escolhido para toda a imagem, com sete métodos de limiarização local, nos quais o valor de limiar é adaptado com base na vizinhança de cada pixel.

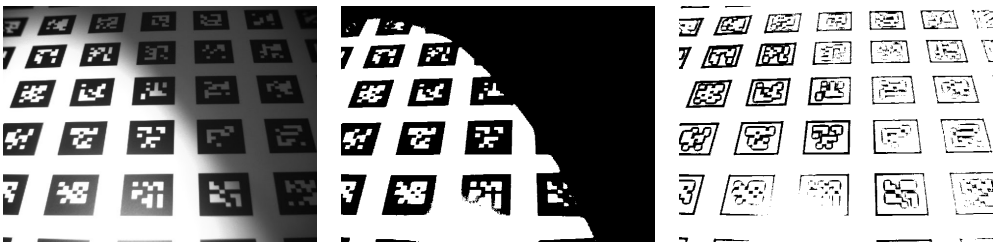
1 Problema

A limiarização é uma técnica de segmentação de imagens, que pode ser usada para particionar pixels em dois conjuntos disjuntos — objeto e fundo — obtendo assim uma imagem resultante bi-nível (preto-e-branco).

A abordagem mais simples para limiarização é a escolha de um único valor de limiar T, a partir do qual transformamos a imagem monocromática $f(x, y)$ na imagem preto-e-branco $g(x, y)$ segundo a relação:

$$g(x,y) = \begin{cases} 0 \text{ (objeto)}, & \text{se } f(x,y) < T \\ 1 \text{ (fundo)}, & \text{se } f(x,y) \geq T \end{cases} \tag{1}$$

A equação 1 caracteriza um método global de limiarização. Entretanto, quando a imagem possui luminosidade não homogênea os métodos globais dificilmente conseguem diferenciar corretamente fundo e objeto, como mostrado na Figura 1).



(a) Imagem original (b) Limiarização global (c) Limiarização local [1]

Figura 1: Comparação de métodos globais e locais de limiarização

Assim, uma forma de melhorar a segmentação de imagens com iluminação não uniforme é adaptar os valores de T de acordo com a vizinhança de cada pixel, levando aos métodos de *threhsolding* local (ou adaptativo).

[1] Método de Sauvola e Pietaksinen, com janela 11×11, $k = 0.5$ e $R = 128$

1.1 Métodos Implementados

Oito métodos foram implementados — sete locais e o método global clássico (1).

Para os métodos locais consideraremos uma janela $w(x, y)$ de tamanho $s \times s$ ao redor de cada pixel. Sendo $p(x, y)$ o pixel na coluna x e linha y da imagem original f, e $q(x, y)$ o pixel correspondente na imagem resultante g, temos:

1.1.1 Global: $q(x, y) = 0$ se $p(x, y) < T$, 1 c.c.

1.1.2 Bernsen: $q(x, y) = 0$ se $p(x, y) < \frac{max(w)+min(w)}{2}$, 1 c.c.

1.1.3 Niblack: $q(x, y) = 0$ se $p(x, y) < \mu(w) + k\sigma(w)$, 1 c.c.

1.1.4 Sauvola e Pietaksinen: $q(x, y) = 0$ se $p(x, y) < \mu(w)\left(1 + k(\frac{\sigma(w)}{R} - 1)\right)$, 1 c.c.

1.1.5 Phansalskar et. al: $q(x, y) = 0$ se $p(x, y) < \mu(w)\left(1 + pe^{-q\mu(w)} + k(\frac{\sigma(w)}{R} - 1)\right)$, 1 c.c.

1.1.6 Contraste: $q(x, y) = 0$ se $|max(w) - p(x, y)| < |min(w) - p(x, y)|$, 1 c.c.

1.1.7 Média: $q(x, y) = 0$ se $p(x, y) < \mu(w)$, 1 c.c.

1.1.8 Mediana: $q(x, y) = 0$ se $p(x, y) < \text{median}(w)$, 1 c.c.

2 Programa

2.1 Dependências e Organização

O trabalho foi desenvolvido em Python `3.7.4`, com as bibliotecas Numpy `1.17.0`, Matplotlib `3.1.1` e OpenCV `4.1.0`, e é dividido em três arquivos:

- `main.py` – tratamento da entrada/saída e execução do projeto
- `thresholding.py` – implementação das técnicas de binarização
- `utils.py` – funções gerais para processamento de imagens

As imagens utilizadas para teste estão na pasta `i/`, e alguns exemplos de saída encontram-se na pasta `o/`. O código também pode ser visto em github.com/laurelkeys/image-processing.

2.2 Execução

As imagens de entrada `.pgm` devem ser colocadas na pasta `i/`, localizada no mesmo diretório do script `main.py`, que salvará as imagens resultantes na pasta `o/` (em `.pgm` por padrão, ou no formato `.png` se a opção `-png` for usada).

Por padrão, aplicam-se todas as técnicas de *thresholding* mostradas na Seção 1 a todas as imagens que estão na pasta `i/`. Entretanto, é possível especificar a imagem e a técnica utilizadas executando-se `main.py -img IMAGE -m {0,1,2,3,4,5,6,7}`.

Pode-se visualizar todas as opções disponíveis com `main.py --help`.

2.3 Valores Padrão

Por padrão, o método global tem *threshold* 128, que pode ser alterado com a flag `-t`, e a janela de vizinhança para os métodos locais é de 3×3 (mas também é possível redefini-la, com a opção `-s`).

Usou-se as constantes recomendadas para:

- Niblack: $k = 0.2$
- Sauvola e Pietaksinen: $k = 0.5$, $R = 128$
- Phansalskar et. al: $k = 0.25$, $R = 0.5$, $p = 2$, $q = 10$

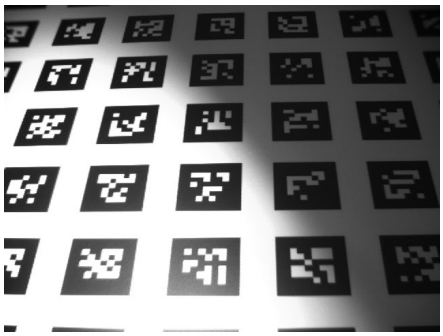
Porém, valores diferentes podem ser especificados passando um arquivo com a opção `-cc`, cujo conteúdo tenha a forma de um dicionário Python, como:

```
{
    "niblack": { "k": -0.2 },
    "sauvola_pietaksinen": { "k": 0.5, "R": 128 },
    "phansalkar_more_sabale": { "k": 0.25, "R": 0.5, "p": 2, "q": 10 }
}
```

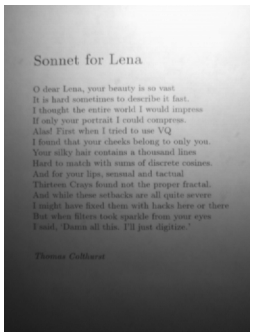
3 Resultados

Os resultados para todas as imagens disponibilizadas podem ser vistos na pasta `o/`.

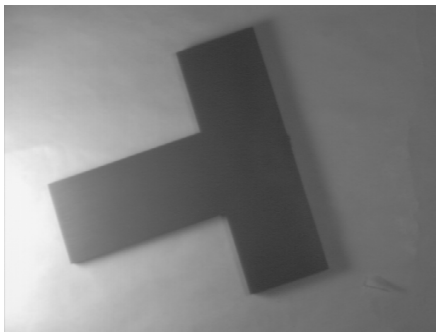
Aqui apresentamos apenas algumas segmentações das imagens `fiducial`, `sonnet` e `wedge` (os resultados mais interessantes), pois estas possuem luminosidade não uniforme e dois tons predominantes (assim, podem ser bem representadas apenas com preto e branco, sem necessitar de um degradê), tornando-as boas possíveis candidatas para a binarização.



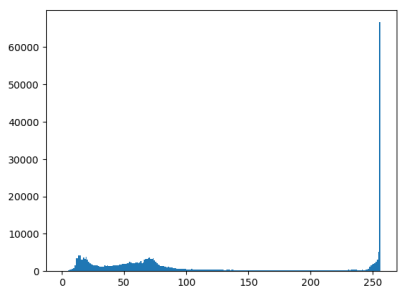
(a) `fiducial.pgm`



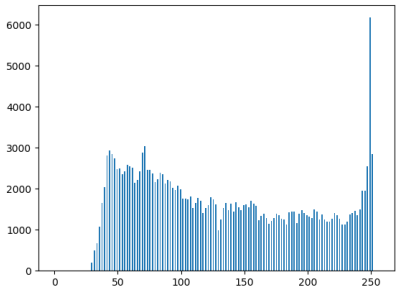
(b) `sonnet.pgm`



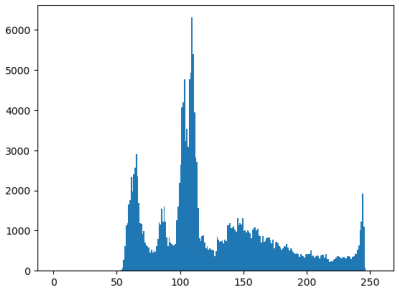
(c) `wedge.pgm`



(d) `fiducial`



(e) `sonnet`



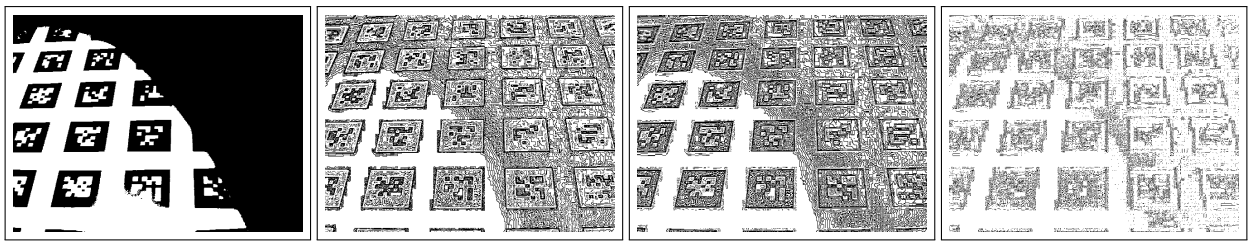
(f) `wedge`

Figura 2: Imagens originais utilizadas (primeira linha) com seus histogramas de níveis de cinza (segunda linha)

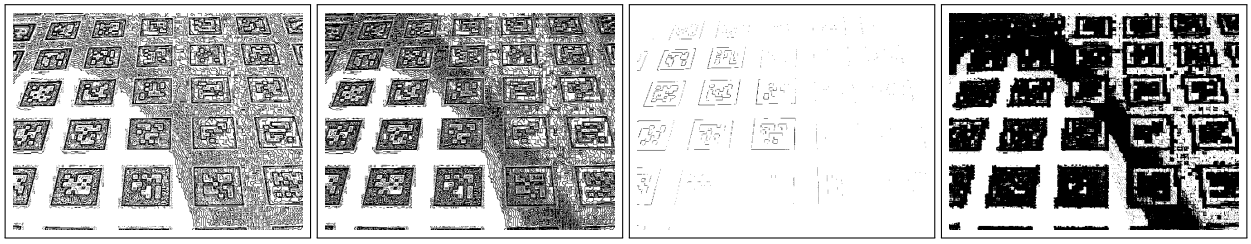
Foram utilizados os valores definidos na Seção 2.3 nos resultados apresentados, a não ser onde mencionado o contrário.

Na legenda das figuras é mostrado a porcentagem de pixels pretos em relação ao total, ou seja, a fração $\frac{\text{n° pixels pretos}}{\text{n° pixels pretos + n° pixels brancos}}$ como as imagens limiarizadas são binárias.

3.1 `fiducial`

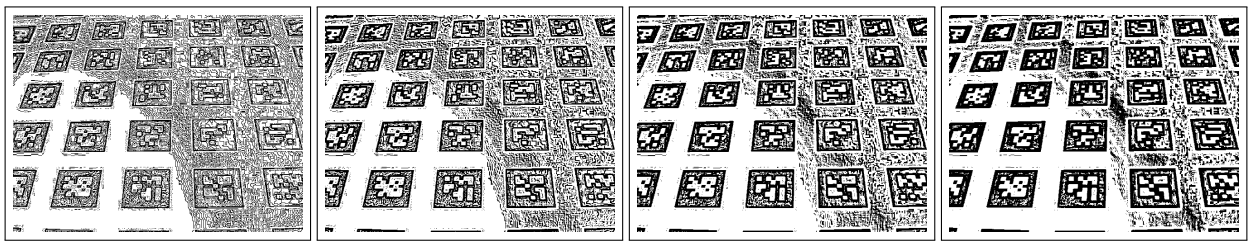


(a) Global (61.85%) (b) Contraste (33.56%) (c) Média (35.38%) (d) Mediana (14.76%)

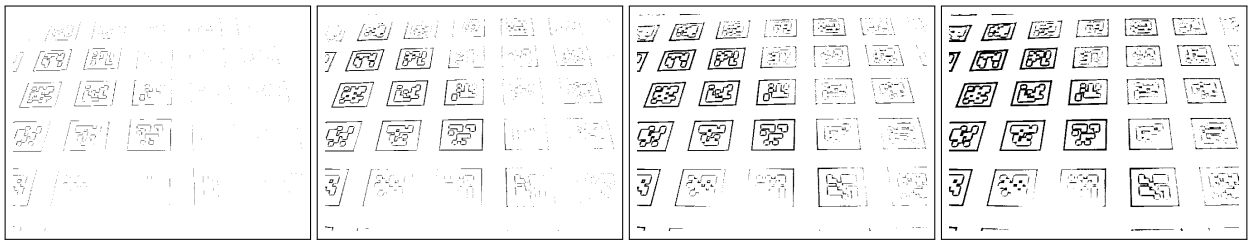


(e) Bersen (33.02%) (f) Niblack (42.67%) (g) Sauvola e Pietaksinen (1.08%) (h) Phansalskar et. al (59.87%)

Figura 3: Resultados da limiarização da imagem `fiducial` com diferentes métodos e porcentagem de pixels pretos



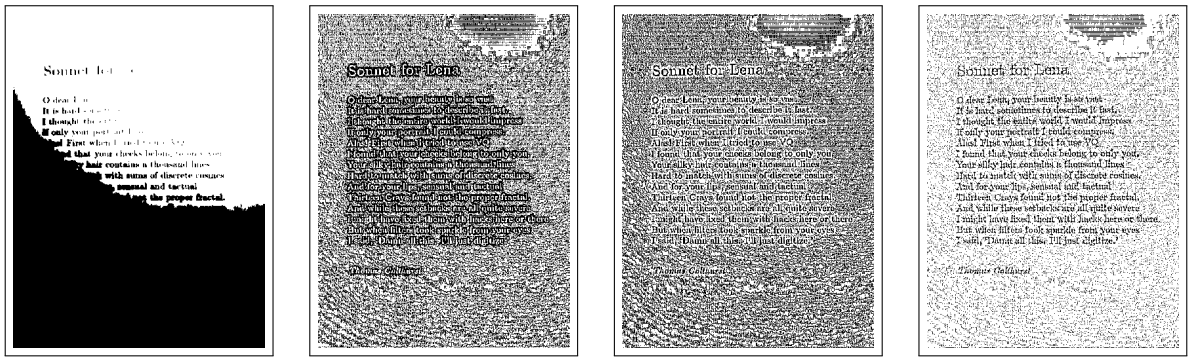
(a) 3×3 (b) 5×5 (c) 7×7 (d) 9×9



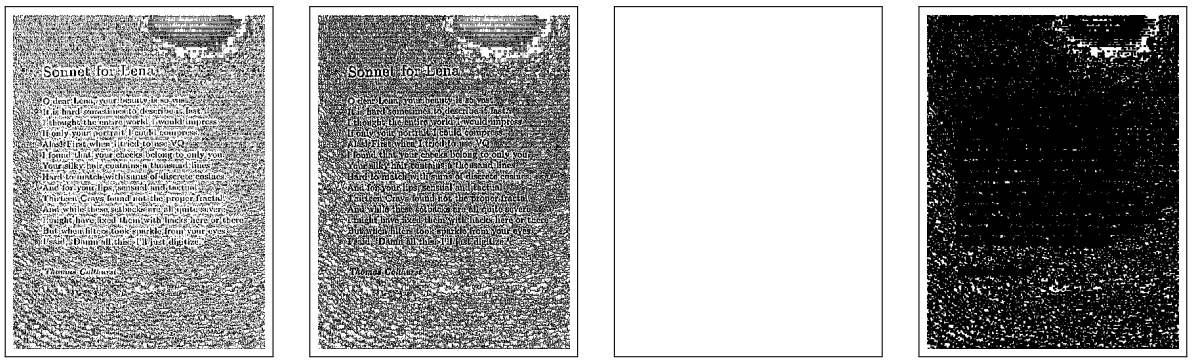
(e) 3×3 (f) 5×5 (g) 7×7 (h) 9×9

Figura 4: Resultados dos métodos de Bersen (primeira linha) e de Sauvola e Pietaksinen (segunda linha) com diferentes janelas de vizinhança

3.2 sonnet



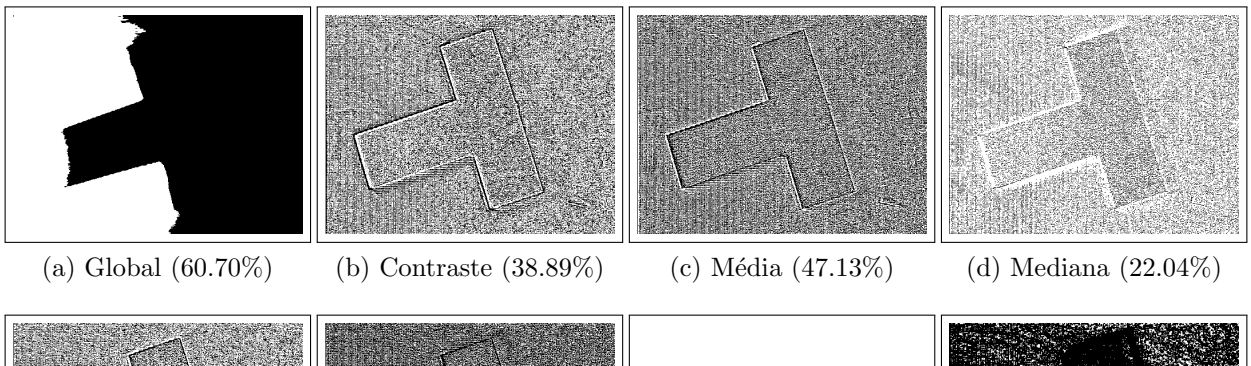
(a) Global (52.06%) (b) Contraste (45.51%) (c) Média (40.60%) (d) Mediana (15.94%)



(e) Bersen (34.27%) (f) Niblack (49.79%) (g) Sauvola e Pietaksinen (0.00%) (h) Phansalskar et. al (92.62%)

Figura 5: Resultados da limiarização da imagem `sonnet` com diferentes métodos e porcentagem de pixels pretos

3.3 wedge



(a) Global (60.70%) (b) Contraste (38.89%) (c) Média (47.13%) (d) Mediana (22.04%)

(e) Bersen (38.86%) (f) Niblack (59.04%) (g) Sauvola e Pietaksinen (0.00%) (h) Phansalskar et. al (86.76%)

Figura 6: Resultados da limiarização da imagem `wedge` com diferentes métodos e porcentagem de pixels pretos

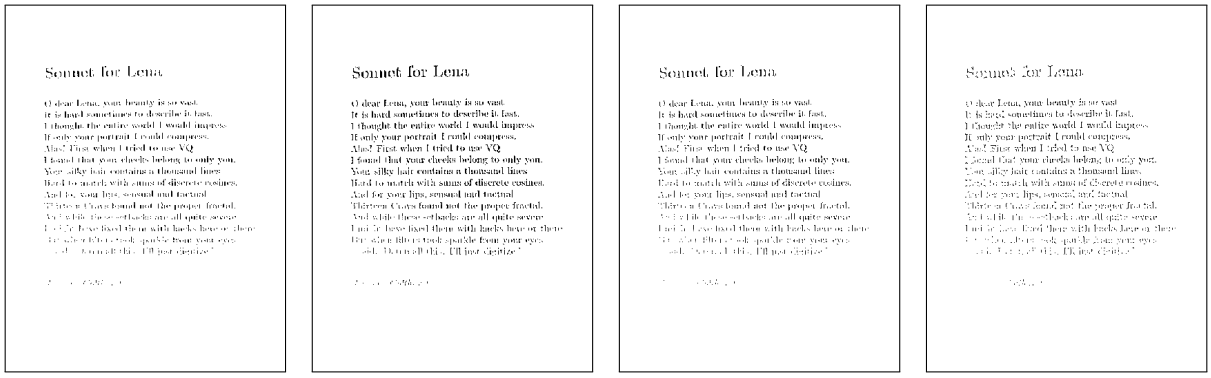
4 Análise e Discussão

Como esperado, vemos claramente nas Figuras 3, 5 e 6 que o método global é inadequado para imagens com sombras.

Entretanto, mesmo os resultados dos métodos adaptativos apresentaram grande quantidade de ruído nas regiões homogêneas das imagens (principalmente na `sonnet`), apesar de terem sucedido em agrupar o conteúdo de interesse em uma mesma classe (p.e. marcadores, texto, e contorno).

Isso ocorre porque as imagens exploradas possuem regiões bem distinguíveis, o que leva a pixels com valores bem diferentes em vizinhanças que contém bordas. Por outro lado, nas regiões uniformes, a média da área da janela não é adequada como um limite, pois a faixa de valores de intensidade em uma vizinhança local é muito pequena (i.e. os valores são todos muito próximos).

Assim, uma possível solução é "deslocar" o valor do pixel central por uma constante C para compará-lo com o *threshold* calculado (i.e. mapeia-se $p(x, y) \rightarrow p(x, y) + C$ antes de calcular $q(x, y)$ conforme as equações da Seção 1.1). Deste modo, pixels em uma vizinhança uniforme são segmentados para a mesma classe [1]. O efeito de utilizar-se isso para a imagem `sonnet` é mostrado na Figura 7.



(a) Método de Bersen (b) Método de Niblack (c) Método da Média (d) Método da Mediana

Figura 7: Resultados da limiarização da imagem `sonnet` para diferentes métodos, aplicando deslocamento $C = 8$ antes da segmentação

A imagem que teve a melhor limiarização foi a `fiducial`, e vemos na Figura 4 que o aumento da janela de 3×3 para 9×9 melhora os resultados do método de Sauvola e Pietaksinen, porém no de Bersen utilizar 7×7 parece ser melhor que 9×9. Valores muito maiores que 9 para a janela de vizinhança pioraram os resultados de todos os métodos.

Por fim, nota-se que os métodos de Sauvola e Pietaksinen e de Phansalskar et. al não segmentaram bem as imagens (exceto o primeiro para a imagem `fiducial`). Contudo, esses são os métodos mais complexos e com mais parâmetros, portanto um ajuste fino dos valores de k e R devem melhorar consideravelmente a limiarização.

5 Conclusões

Pelos resultados apresentados constatamos que métodos de limiarização local/adaptativa são capazes de segmentar imagens de cenas com dois elementos de tons distintos (p.e. textos e documentos) mesmo com iluminação variável, enquanto o método global de escolher-se apenas um *threshold* para toda a imagem não discerne as áreas sombreadas.

Vemos que os métodos locais mais complexos (de Sauvola e Pietaksinen e de Phansalskar et. al) não mostraram resultados melhores, pelo contrário, eles ficaram entre os piores. A limiarização feita com o simples método de Bersen foi superior a ambos (usando-se uma janela 3×3). Percebemos então que, apesar da maior quantidade de parâmetros possibilitar um ajuste fino para as imagens tratadas (levando presumivelmente a resultados melhores que os métodos mais simples), tal ajuste demanda tempo e diversas tentativas, além de não ser generalizável para outras imagens.

Logo, quando não temos grande conhecimento sobre o tipo de imagens que devem ser segmentadas, ou se não temos o tempo de ajustar os parâmetros para cada imagem que iremos tratar, os métodos locais são preferíveis e apresentam resultados aceitáveis, principalmente com a mudança sugerida na Seção 4 de aplicar-se um "deslocamento" nos valores dos pixels antes de compará-los ao valor de limiar.

Referências

[1] A. W. Robert Fisher, Simon Perkins and E. Wolfart, "Adaptive Thresholding," 2003, acessado em setembro de 2019. [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/HIPR2/adpthrsh.htm 6