Frequency analysis in images
2D Fourier Transform

Revision: 1.9, dated: April 30, 2008

Tomáš Svoboda

Czech Technical University, Faculty of Electrical Engineering
Center for Machine Perception, Prague, Czech Republic

svoboda@cmp.felk.cvut.cz

http://cmp.felk.cvut.cz/~svoboda

Frequency Analysis

Decomposition of signals into sequences of sinusoids $\cos, \sin$.

Why?

Frequency Analysis

Decomposition of signals into sequences of sinusoids $\cos, \sin$.

Why?

- Speech, music analysis.

Frequency Analysis

Decomposition of signals into sequences of sinusoids $\cos, \sin$.

Why?

- Speech, music analysis.
- Filtering: boosting or attenuating of specific frequencies.

Frequency Analysis

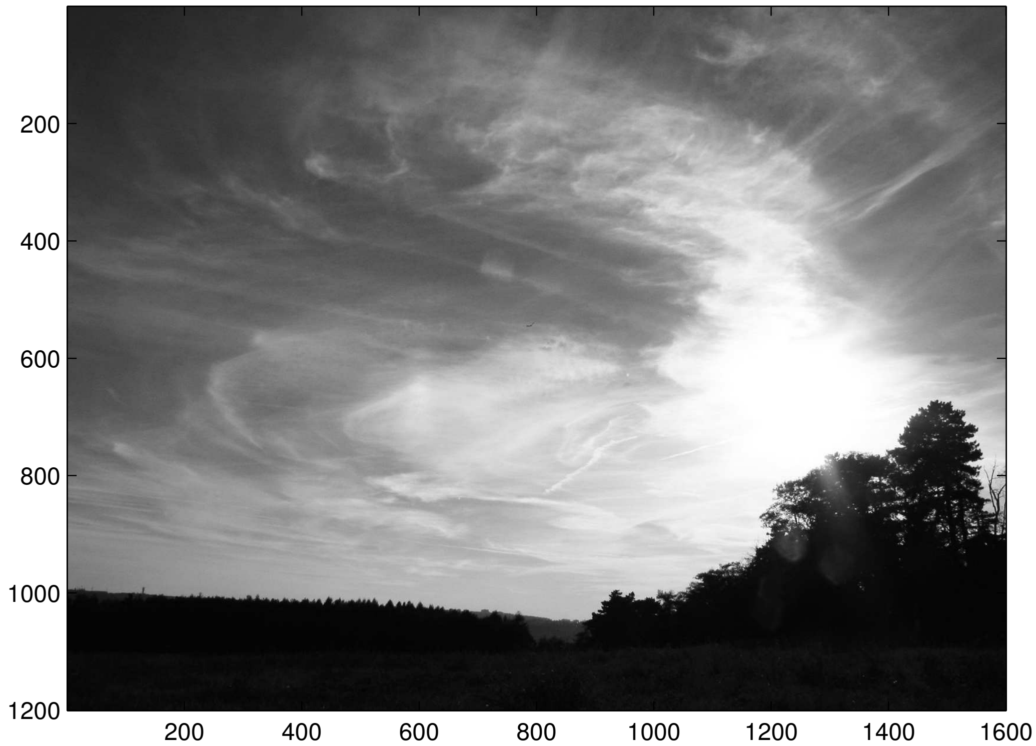
Decomposition of signals into sequences of sinusoids $\cos, \sin$.

Why?

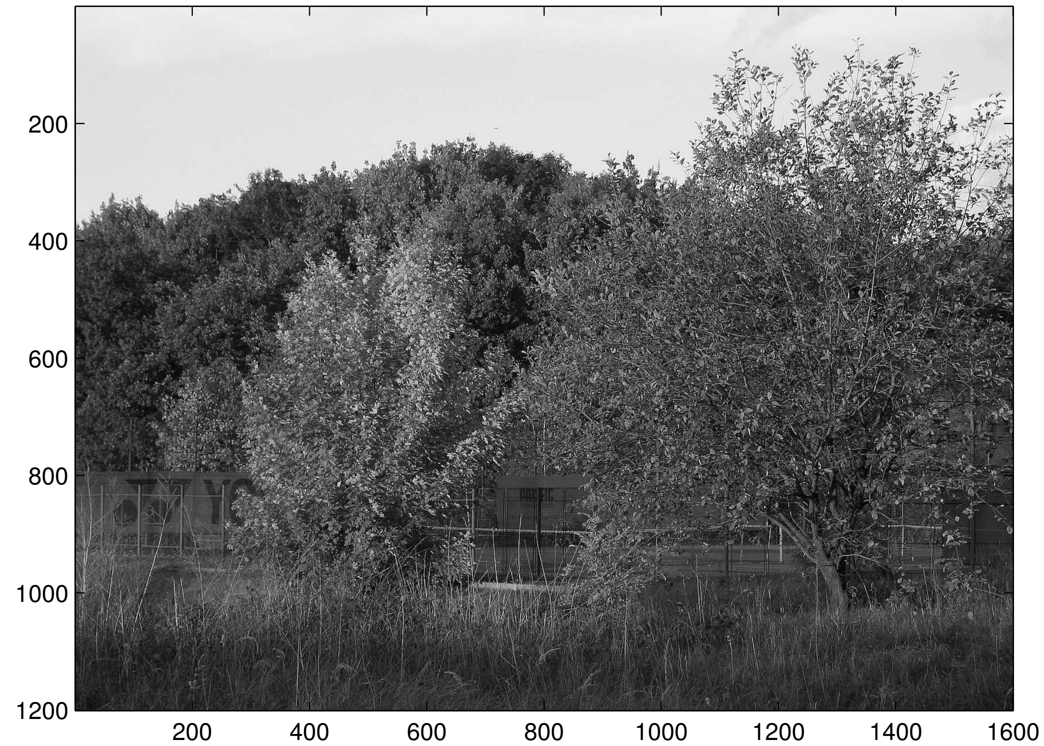
- Speech, music analysis.
- Filtering: boosting or attenuating of specific frequencies.
- Some image distortions may be characterized well in frequency domain.

What are frequencies in real images

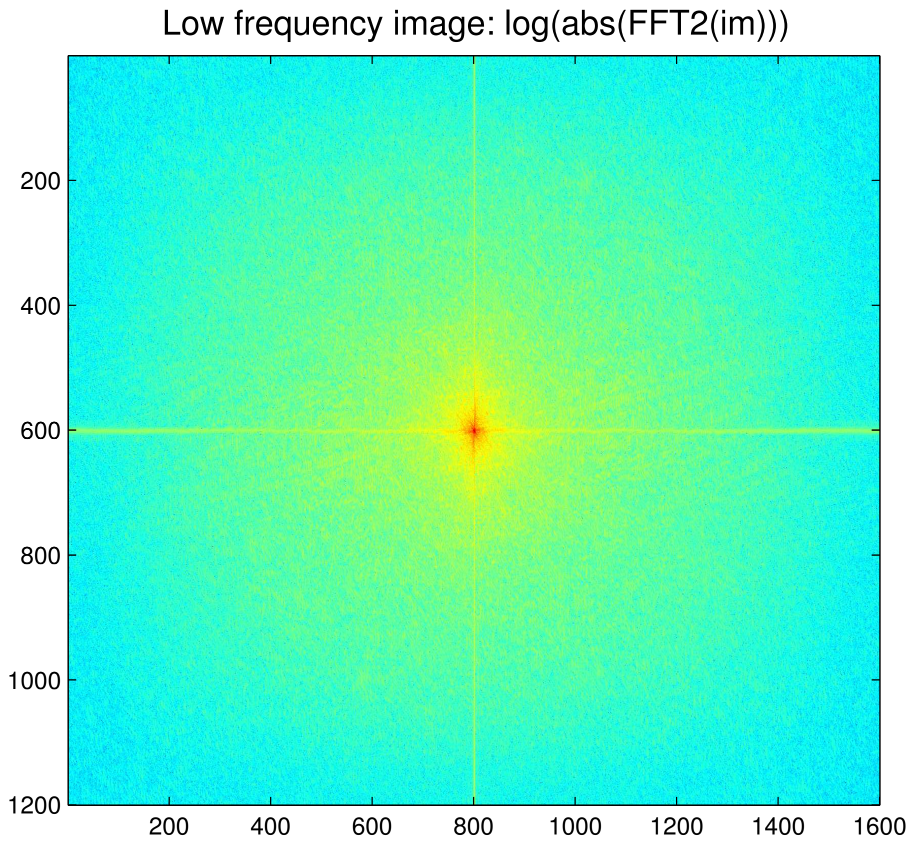
Low frequency image in gray scales

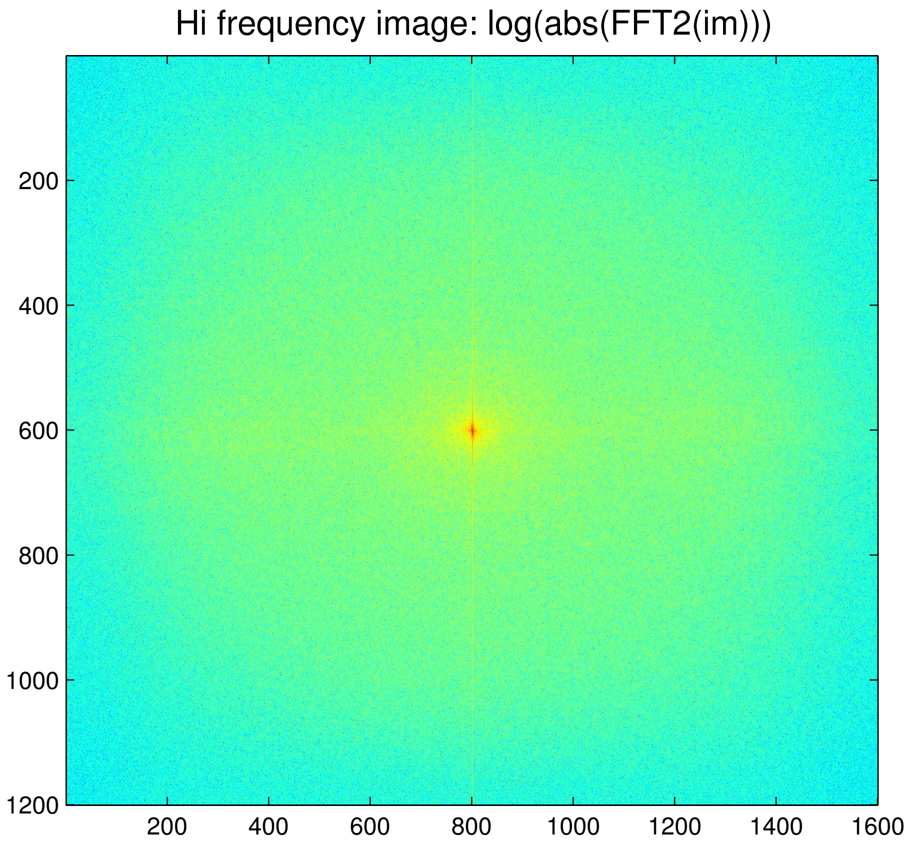


Hi frequency image in gray scales



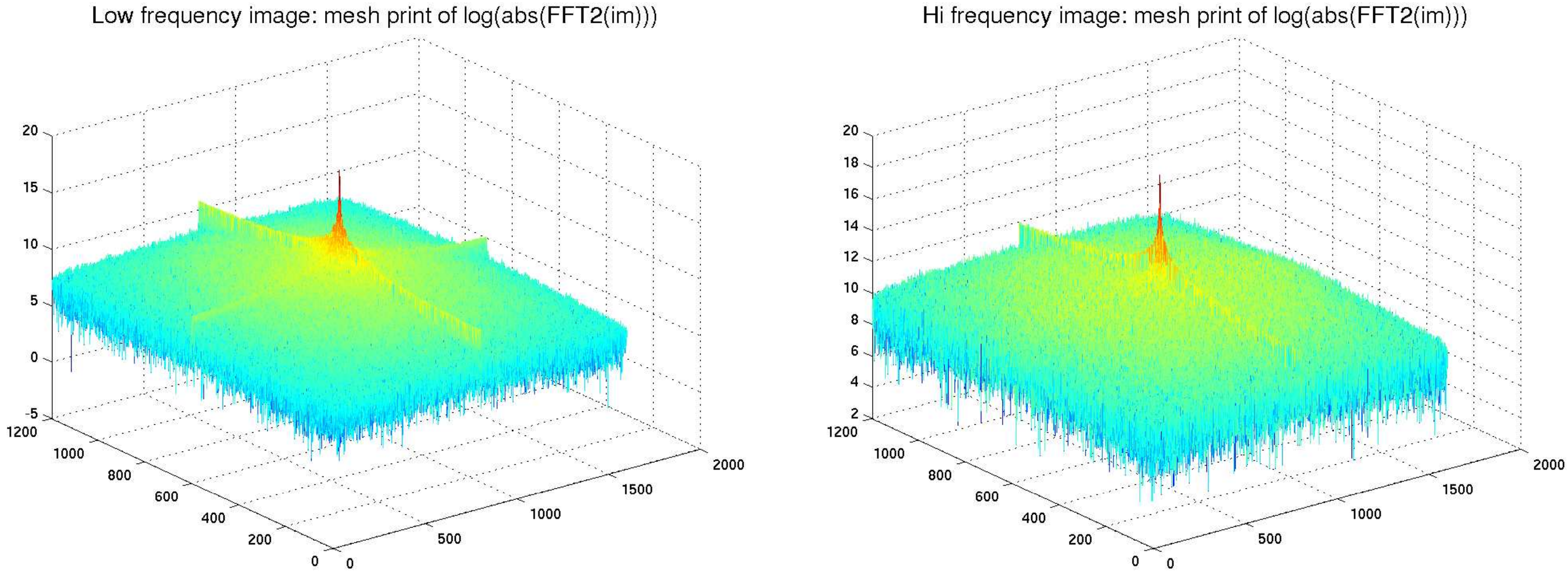
What are frequencies in real images



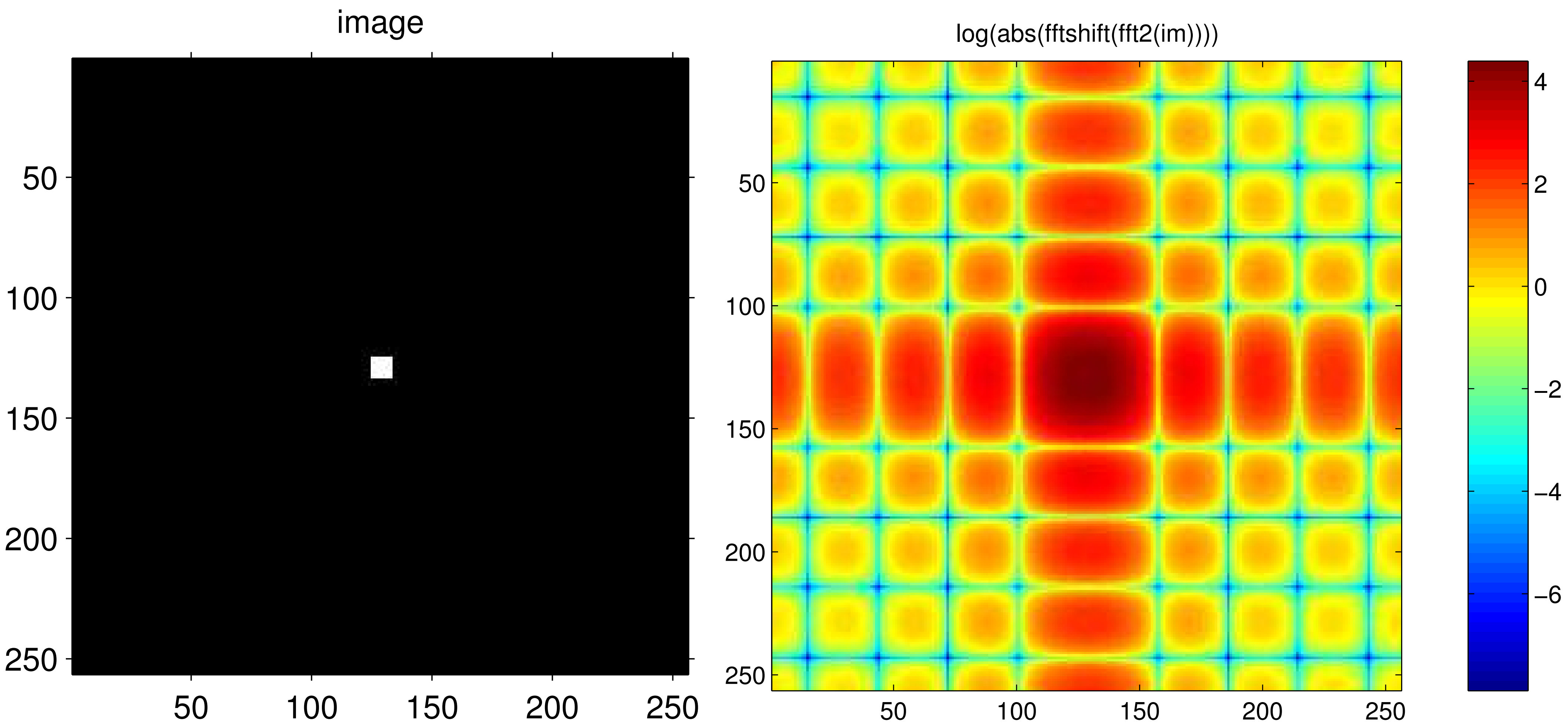


The further from image center, $[600, 800]$, the higher frequency. Do not care about deatils we will come to this later.

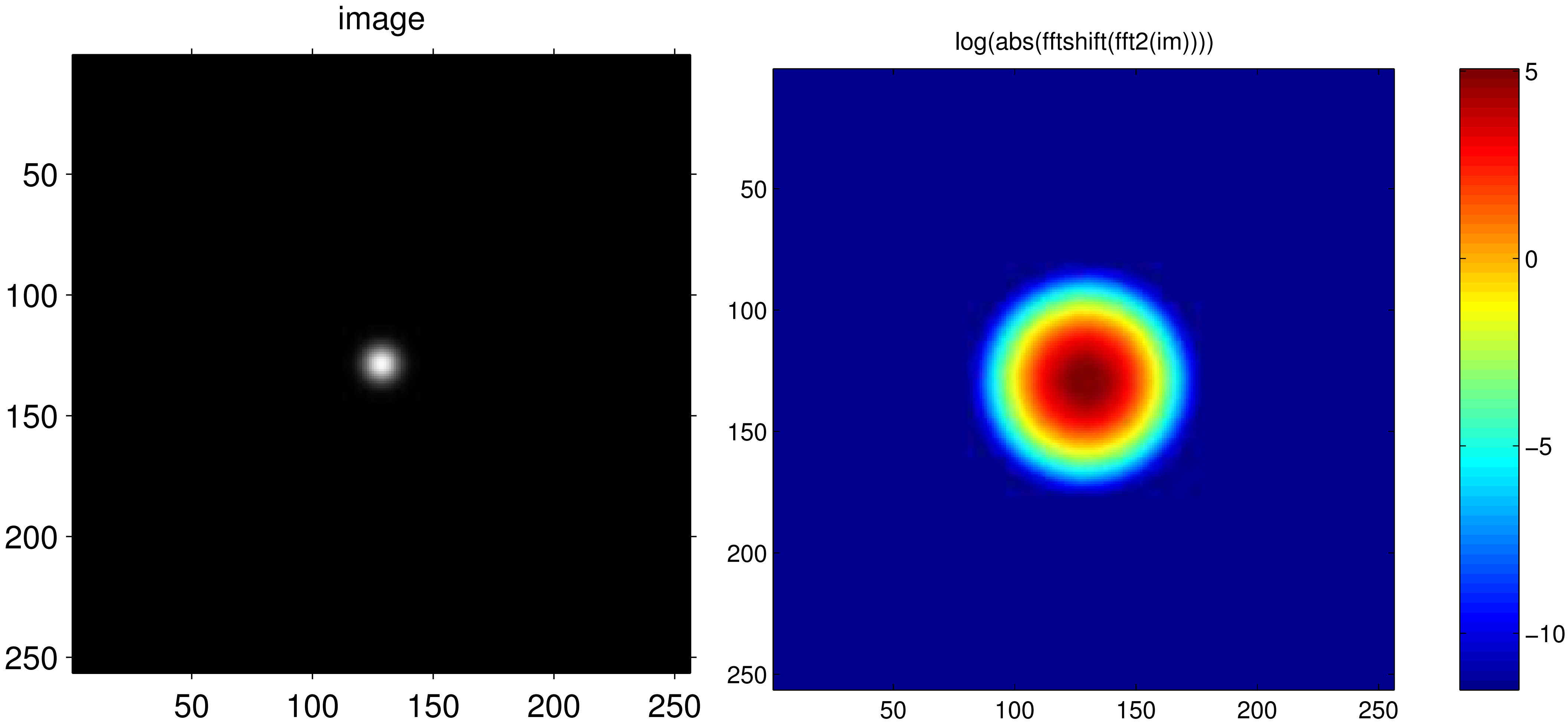
What are frequencies in real images



What are frequencies in artificial images



What are frequencies in artificial images



Fourier transform — Decomposition into sinusoids

Mathematical procedure for computing how is each frequency u in the input signal $f(x)$.

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi ux} dx$$

Fourier transform — Decomposition into sinusoids

Mathematical procedure for computing how is each frequency u in the input signal $f(x)$.

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi ux} dx$$

and back to image/time space

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{i2\pi xu} du$$

Fourier transform — Decomposition into sinusoids

Mathematical procedure for computing how is each frequency u in the input signal $f(x)$.

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi ux} dx$$

and back to image/time space

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{i2\pi xu} du$$

do not forget that

$$\begin{aligned} e^{-i2\pi ux} &= \cos(2\pi ux) - i\sin(2\pi ux) \, , \\ e^{i2\pi ux} &= \cos(2\pi ux) + i\sin(2\pi ux) \, . \end{aligned}$$

2D Fourier transform

Straightforward generalization of the 1D transform

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-i2\pi(ux+vy)}dxdy$$

$$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)e^{i2\pi(xu+yv)}dudv$$

Remark: f has a fourier image F if it is integrable, i.e. has finite energy.

2D FT — Corrugation viewpoint

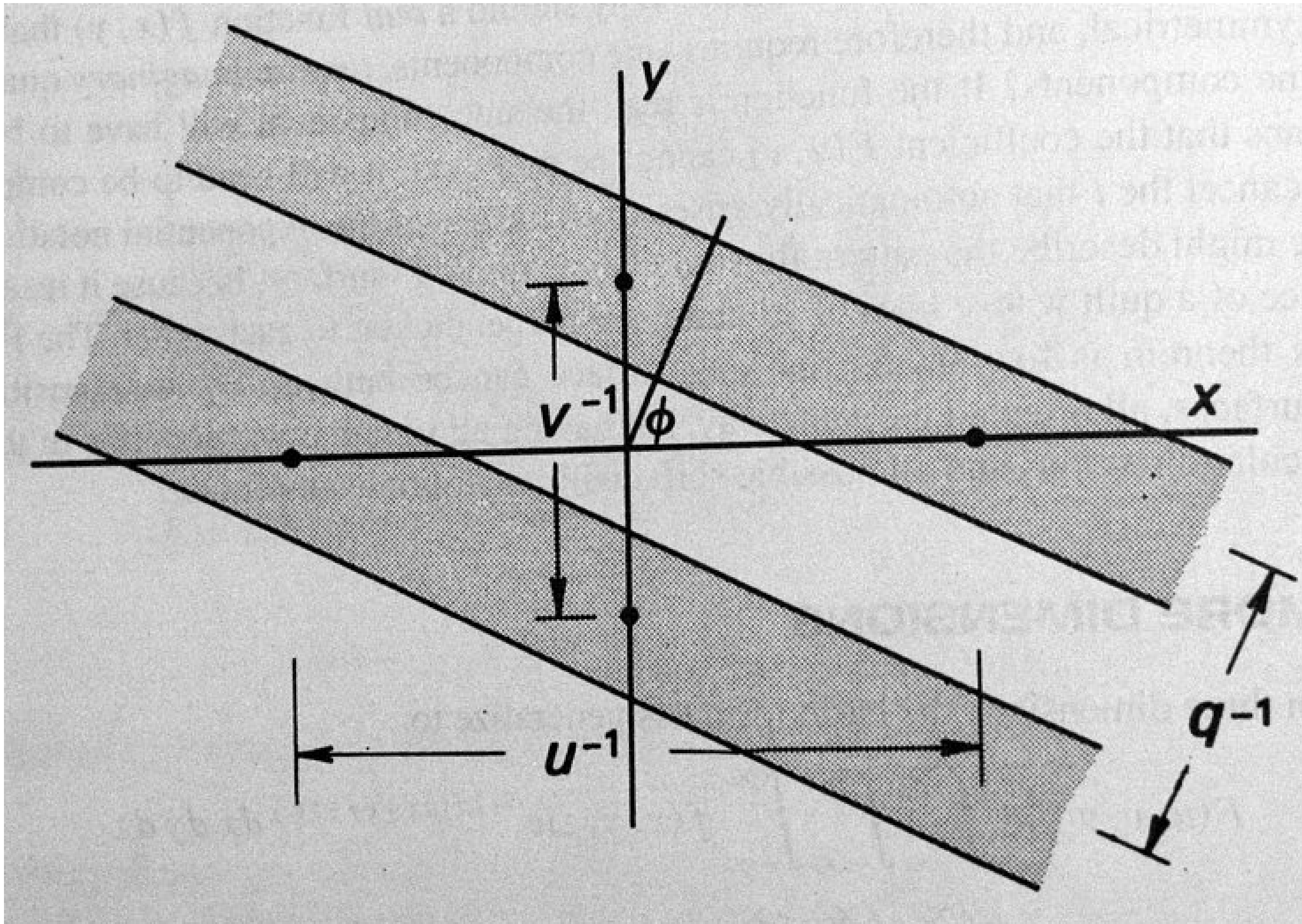


Figure 4-2 A "corrugation" $\cos[2\pi(ux + vy)]$. The shaded zones are negative. The spatial frequency in the x-direction is u (in the y-direction, v); *the* spatial frequency is q.

Illustration taken from [1].

2D FT — Corrugation viewpoint

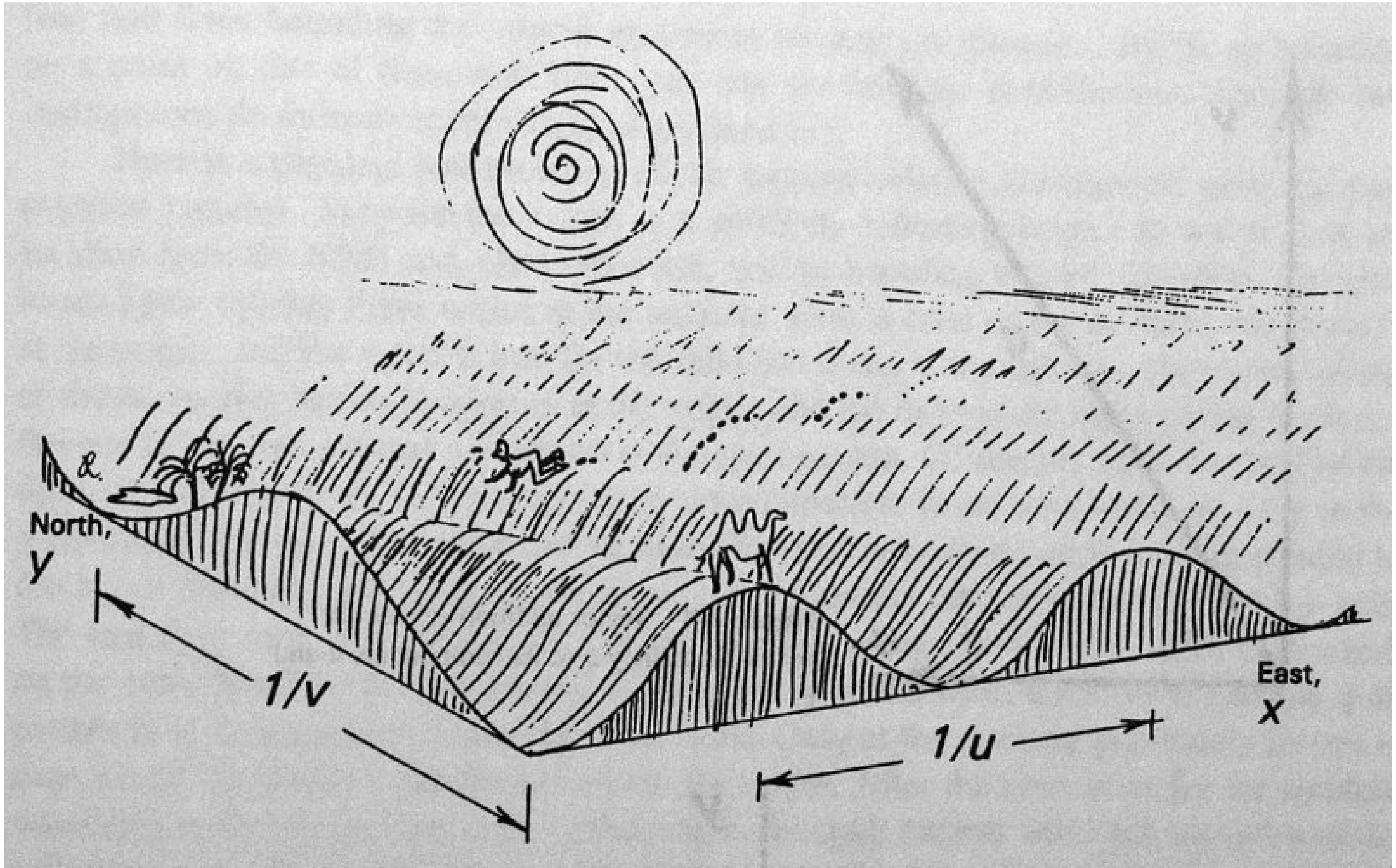


Figure 4-3 The x-component of spatial frequency is u, measured in cycles per unit of x. The spatial period, or crest-to-crest distance traveling east, is u^{-1} (see camel). The spatial frequency q is measured in the direction of hardest going (see man).

Illustration taken from [1].

Discrete Fourier transform

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \exp\left(-i2\pi(\frac{ux}{M} + \frac{vy}{N})\right)$$

Discrete Fourier transform

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \exp\left(-i2\pi(\frac{ux}{M} + \frac{vy}{N})\right)$$

$$f(x,u) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) \exp\left(i2\pi(\frac{ux}{M} + \frac{vy}{N})\right)$$

Discrete Fourier transform

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \exp\left(-i2\pi(\frac{ux}{M} + \frac{vy}{N})\right)$$

$$f(x,u) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) \exp\left(i2\pi(\frac{ux}{M} + \frac{vy}{N})\right)$$

Discrete and finite range of $f(x,y)$, and $F(u,v)$ has some important consequences . . .

Discrete Fourier transform — DC part

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \exp\left(-i2\pi(\frac{ux}{M} + \frac{vy}{N})\right)$$

---

x is an abbreviation for $x_0 + x\delta_x$.

u is $u_0 + u\delta_u$, where $\delta_u = \frac{1}{M\delta_x}$.

Discrete Fourier transform — DC part

$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\exp\left(-i2\pi(\frac{ux}{M}+\frac{vy}{N})\right)$

---

x is an abbreviation for $x_0 + x\delta_x$.

u is $u_0 + u\delta_u$, where $\delta_u = \frac{1}{M\delta_x}$.

DC part:

$F(0,0) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)$

which is the mean of f. But beware of actual implementation! Sometimes it may be only proportional to the mean.

Remeber the DC part, we will need it to understand some filtering results.

Discrete Fourier transform — Periodicity

$F(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(x) \exp\left(-i2\pi(\frac{ux}{M})\right)$

---

x is an abbreviation for $x_0 + x\delta_x$. Hence, $x = 0, 1, \ldots M - 1$

u is $u_0 + u\delta_u$, where $\delta_u = \frac{1}{M\delta_x}$.

Discrete Fourier transform — Periodicity

$F(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(x) \exp\left(-i2\pi(\frac{ux}{M})\right)$

---

x is an abbreviation for $x_0 + x\delta_x$. Hence, $x = 0, 1, \ldots M-1$

u is $u_0 + u\delta_u$, where $\delta_u = \frac{1}{M\delta_x}$.

Periodicity:

We know that: $\exp\left(-i2\pi(\frac{ux}{M})\right) = \cos(2\pi ux/M) - i\sin(2\pi ux/M)$.

Discrete Fourier transform — Periodicity

$F(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(x) \exp\left(-i2\pi(\frac{ux}{M})\right)$

---

x is an abbreviation for $x_0 + x\delta_x$. Hence, $x = 0, 1, \ldots M-1$

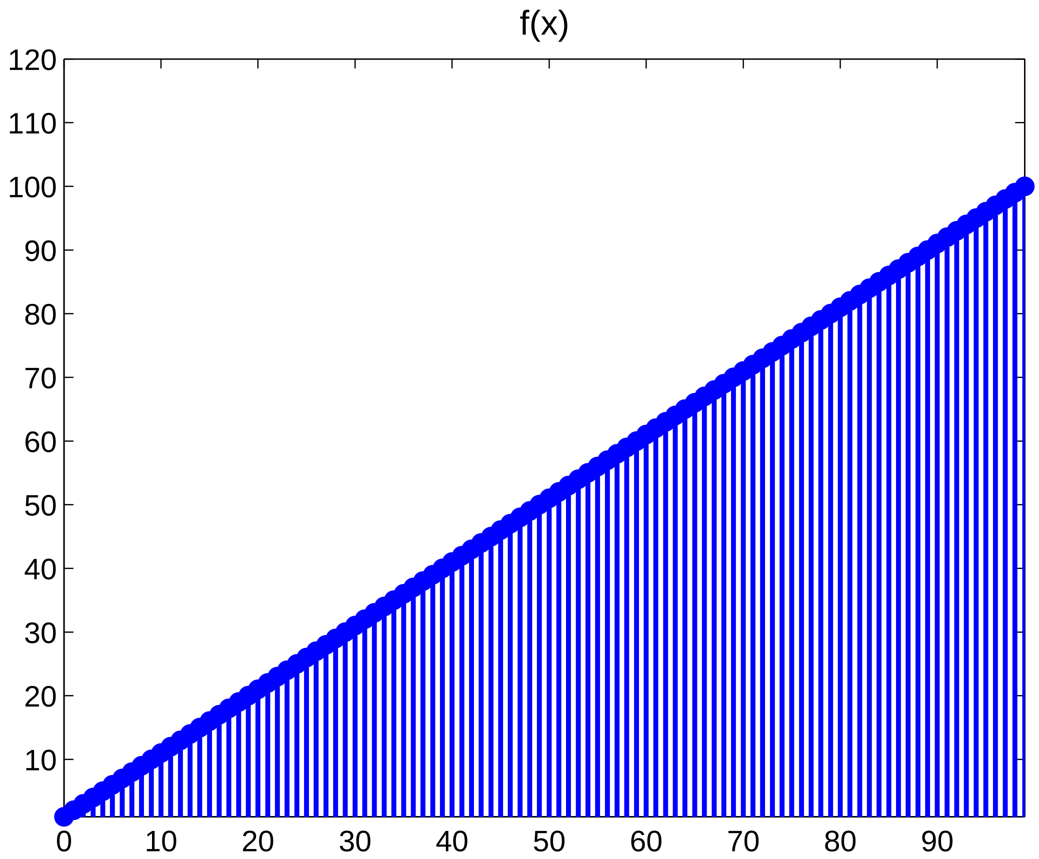
u is $u_0 + u\delta_u$, where $\delta_u = \frac{1}{M\delta_x}$.

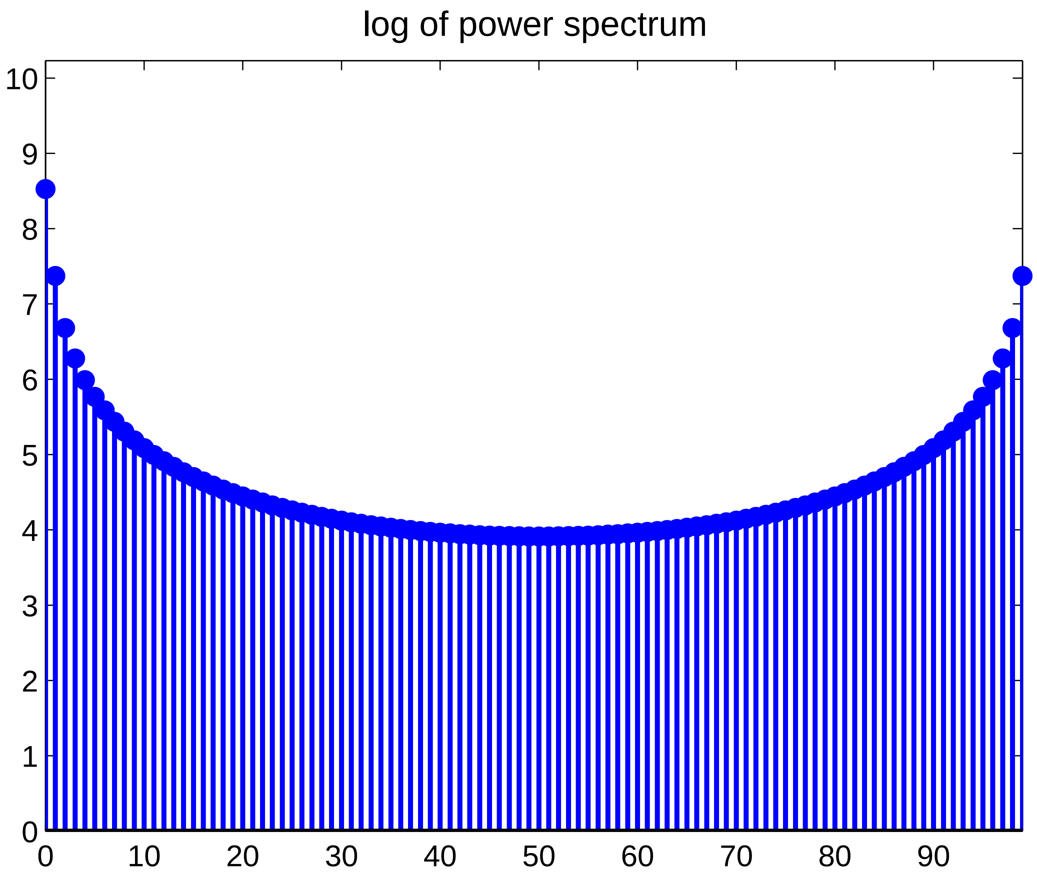
Periodicity:

We know that: $\exp\left(-i2\pi(\frac{ux}{M})\right) = \cos(2\pi ux/M) - i\sin(2\pi ux/M)$.

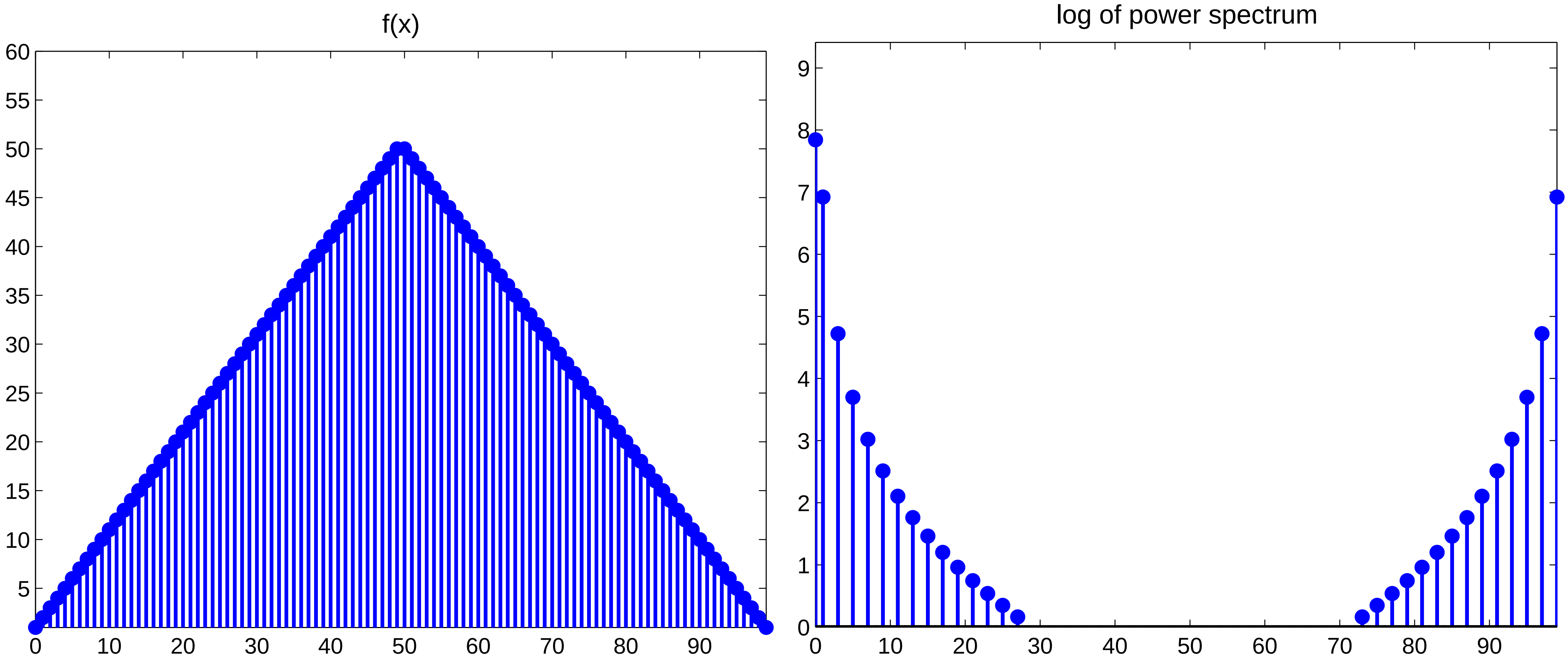
It means that $F(u)$ is periodic with the period M.

Periodicity of Fourier Transform I

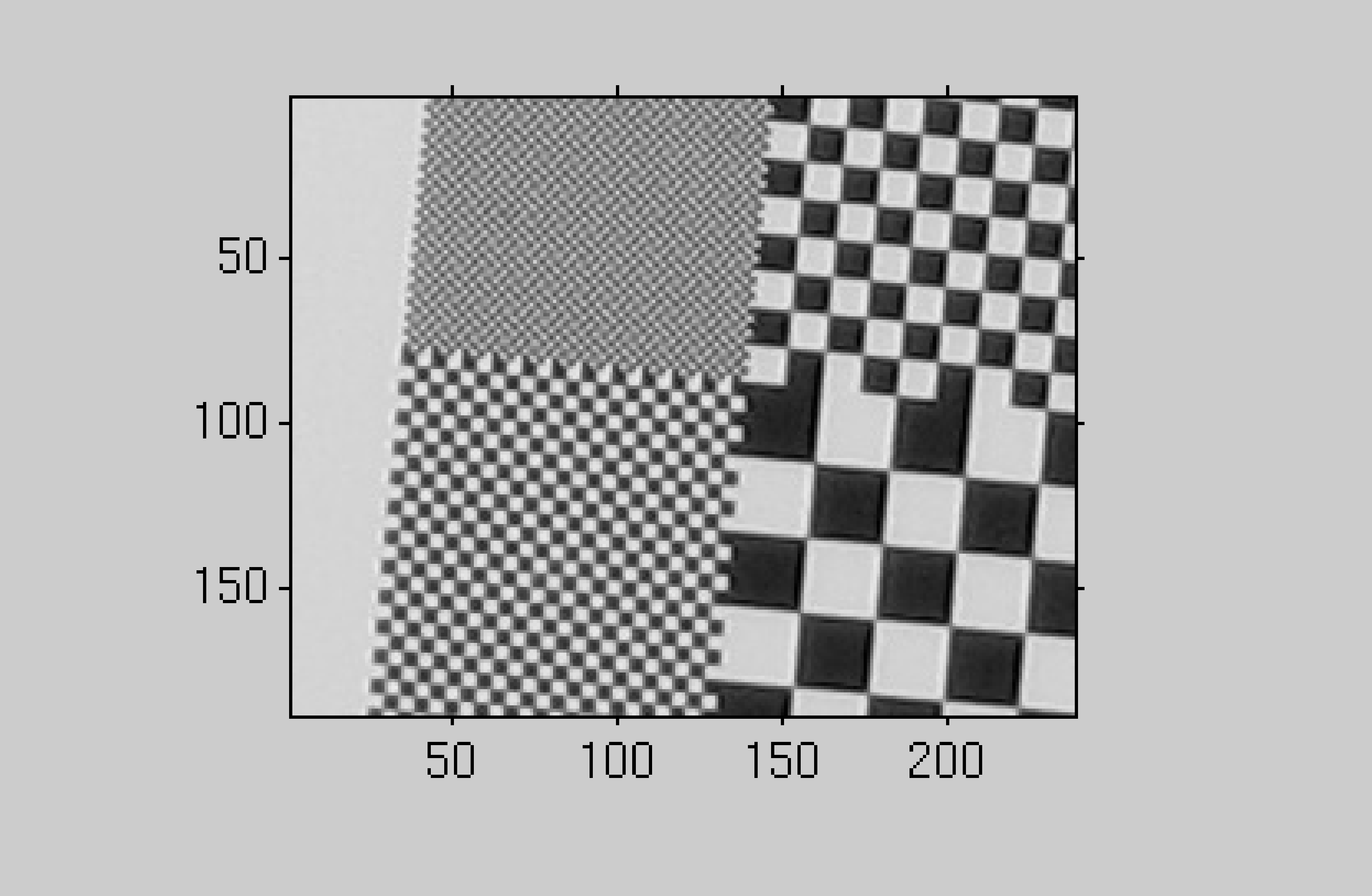




Periodicity of Fourier Transform II

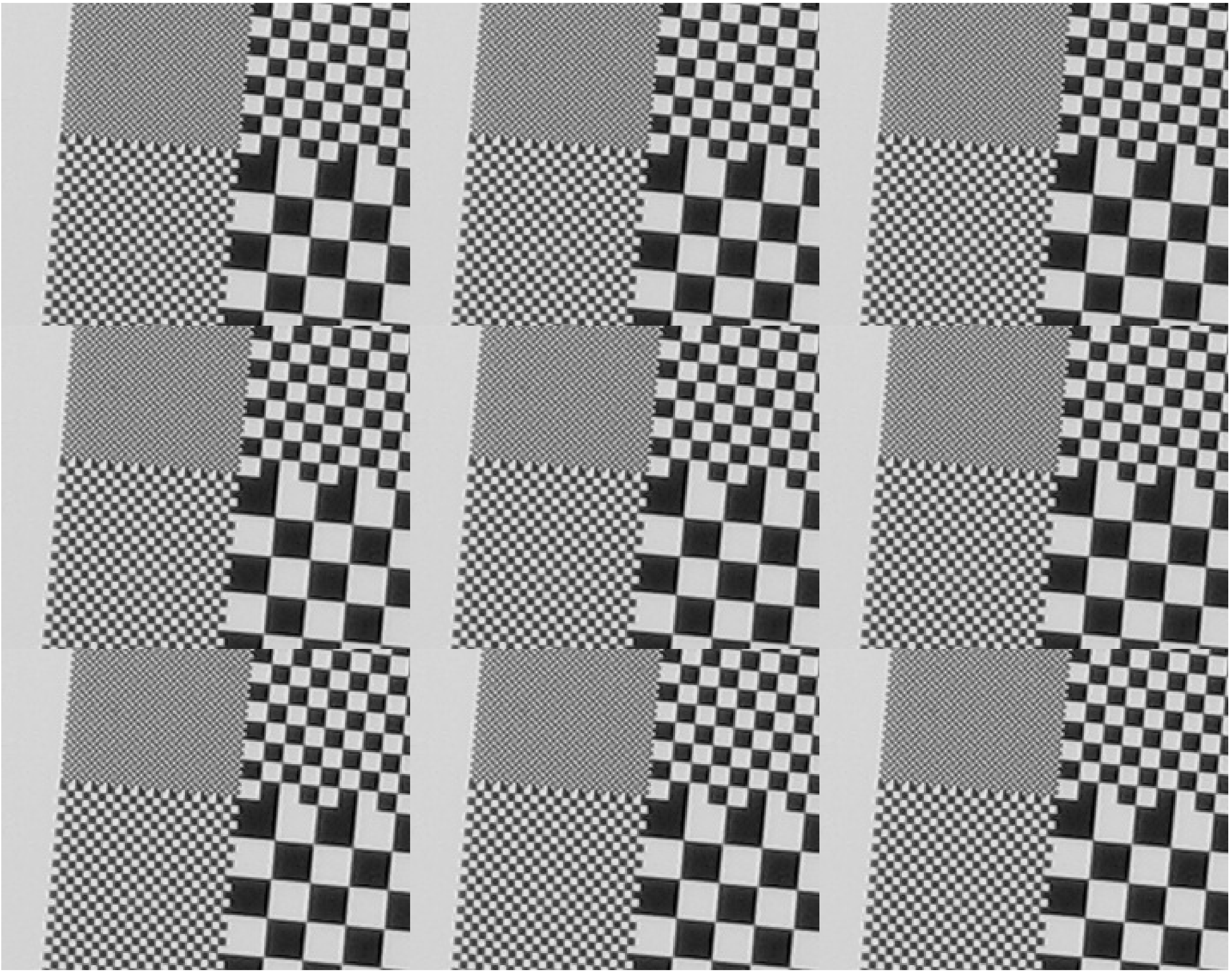


Periodicity of Fourier Transform in Images



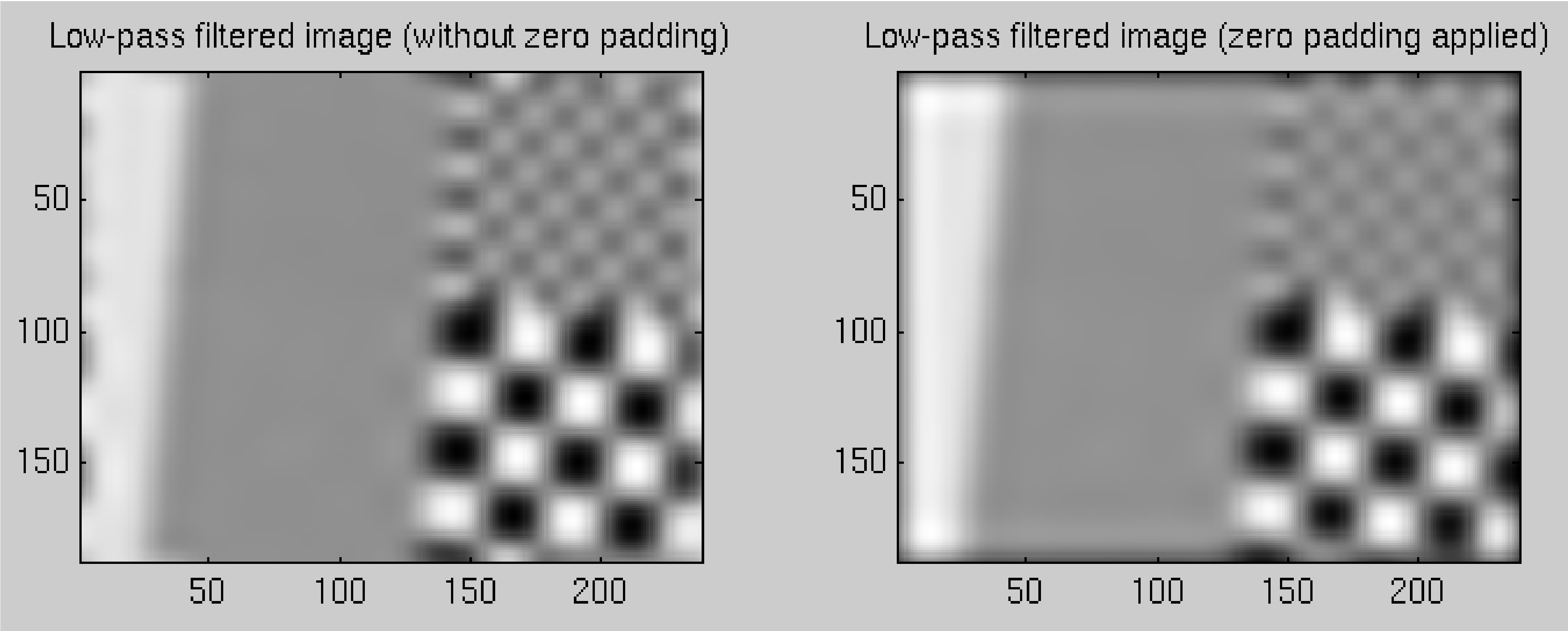
Input image. However, DFT assumes . . .

Periodicity of Fourier Transform in Images – cont.

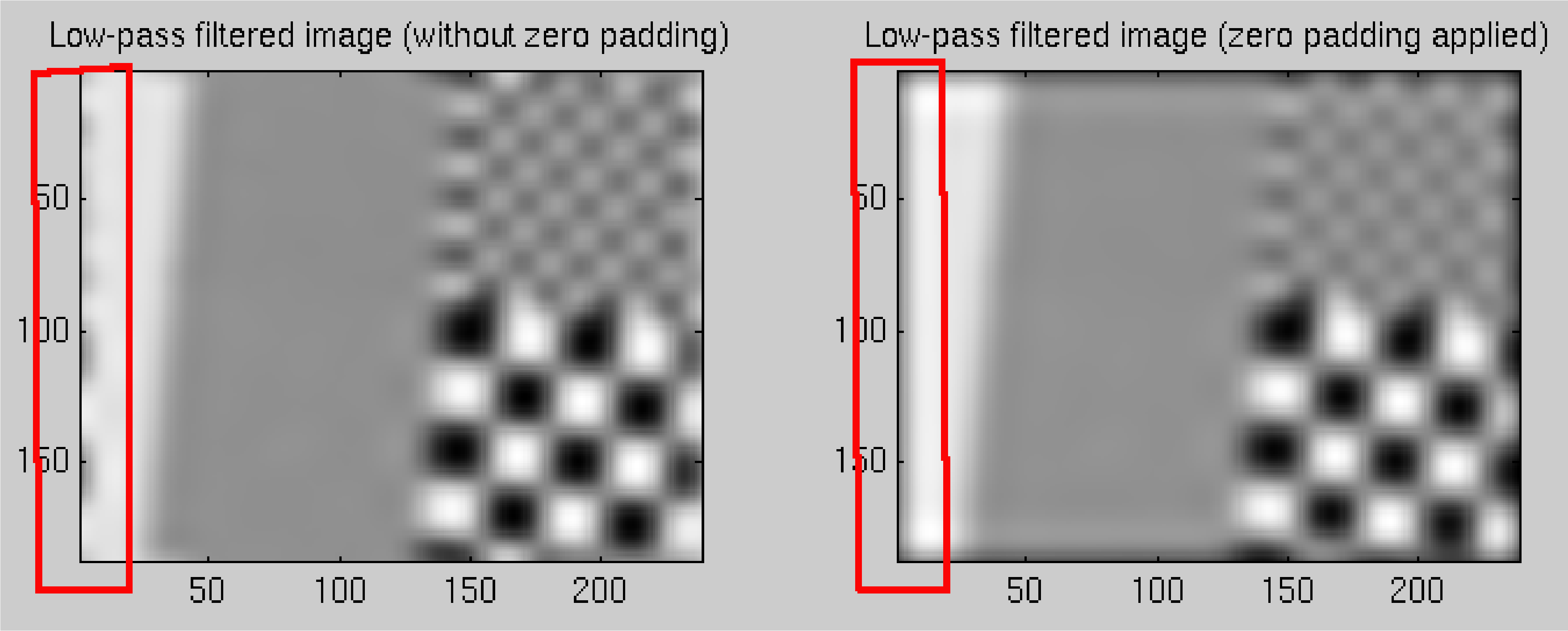


. . . periodicity of the image.

Periodicity of Fourier Transform in Images – cont.



Periodicity of Fourier Transform in Images – cont.



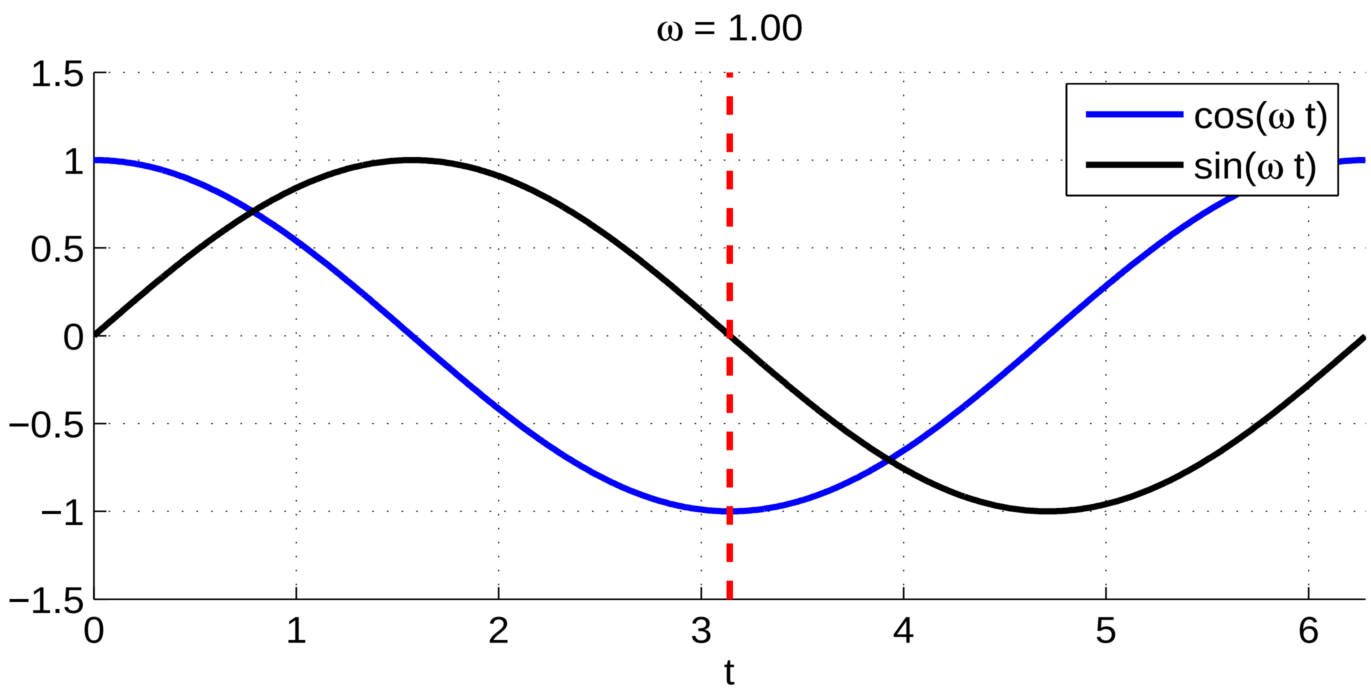
The periodicity is sometimes undesirable, apply zero-padding.

Discrete Fourier transform — conjugate symmetry

$F(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(x) \exp\left(-i2\pi \frac{ux}{M}\right)$ for real $f(x)$.

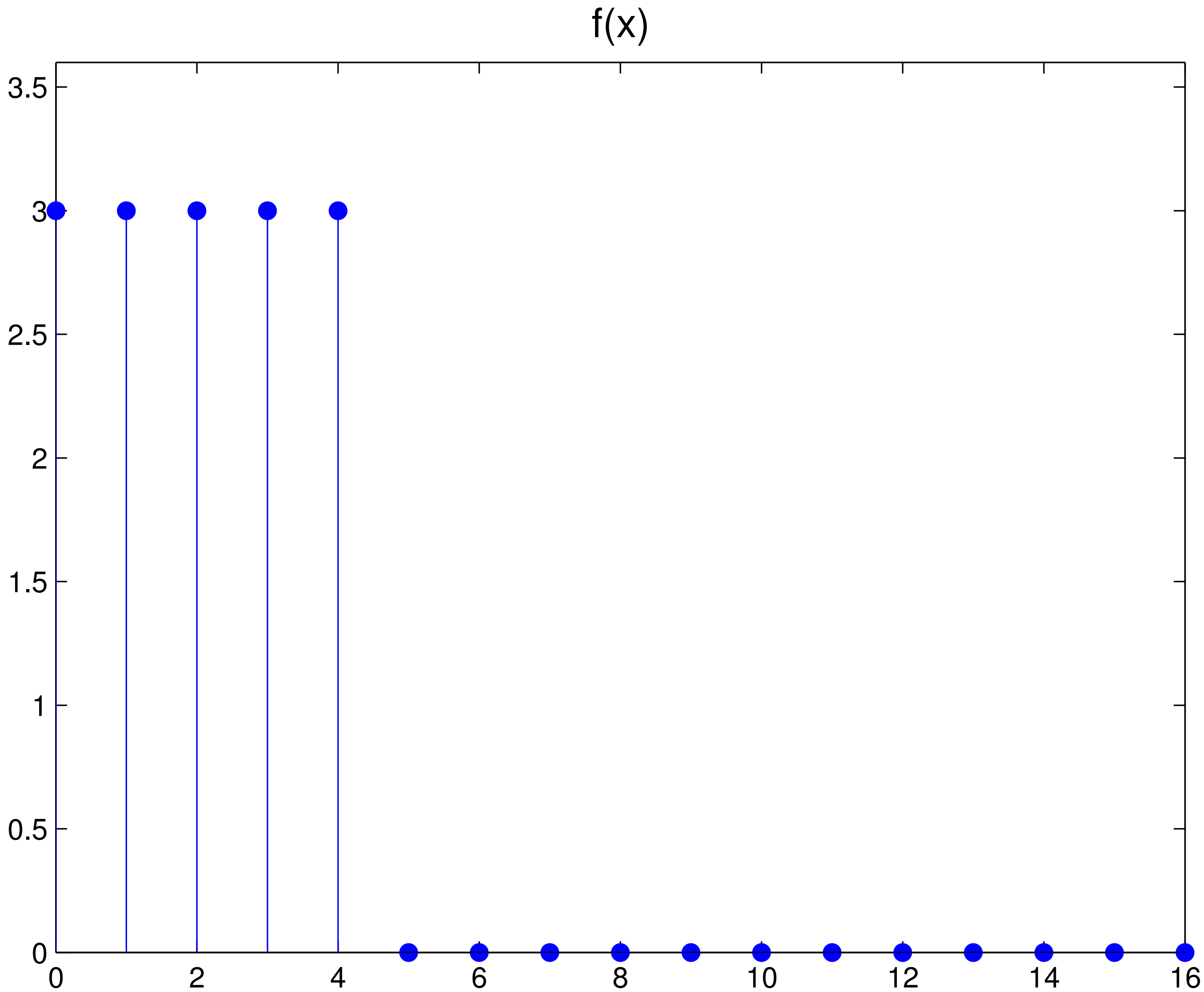
Conjugate symmetry: $F(M-u) = F^*(u)$, $u = 0, \ldots, M-1$.

Remind $e^{-i2\pi ux/M} = \cos(2\pi ux/M) + i \sin(2\pi ux/M)$.

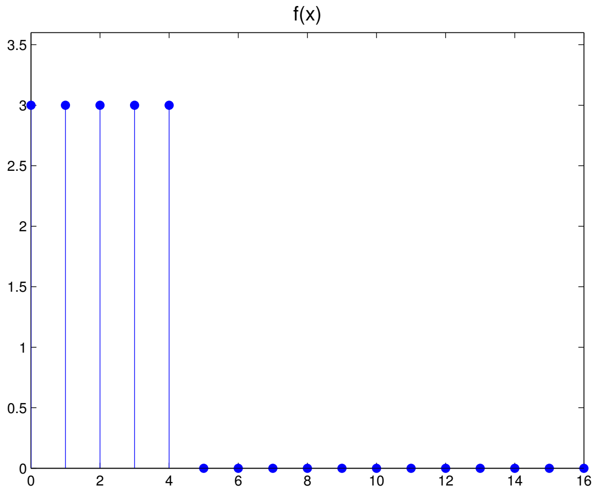


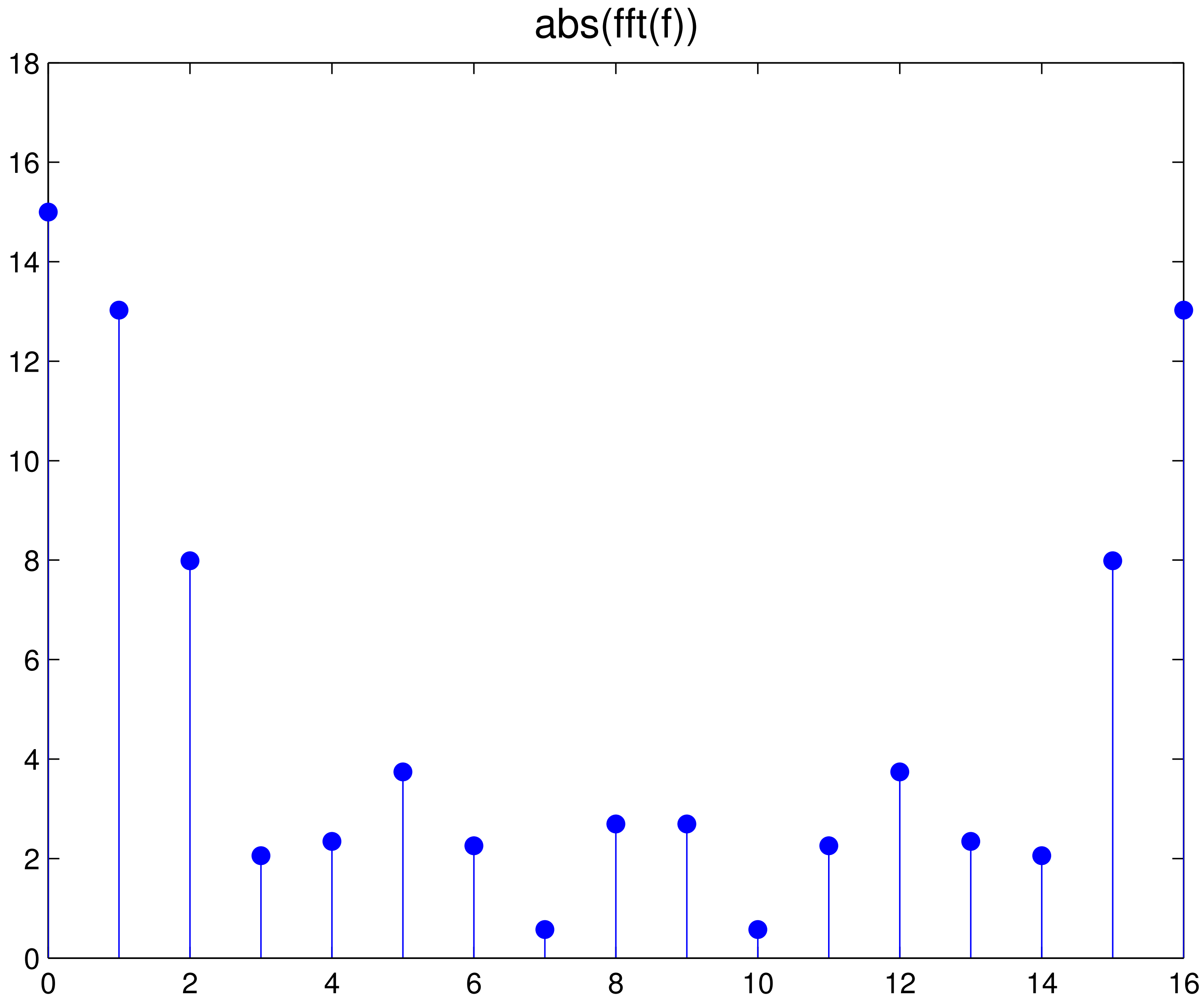
Note that conjugate symmetry implies $|F(M-u)| = |F(u)|$.

Conjugate symmetry in practice – Input

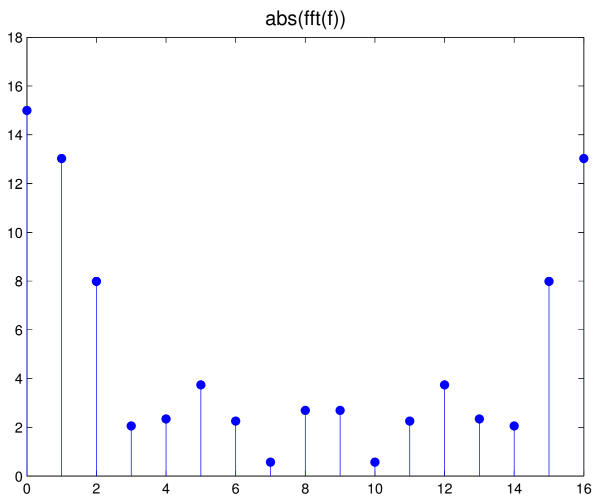


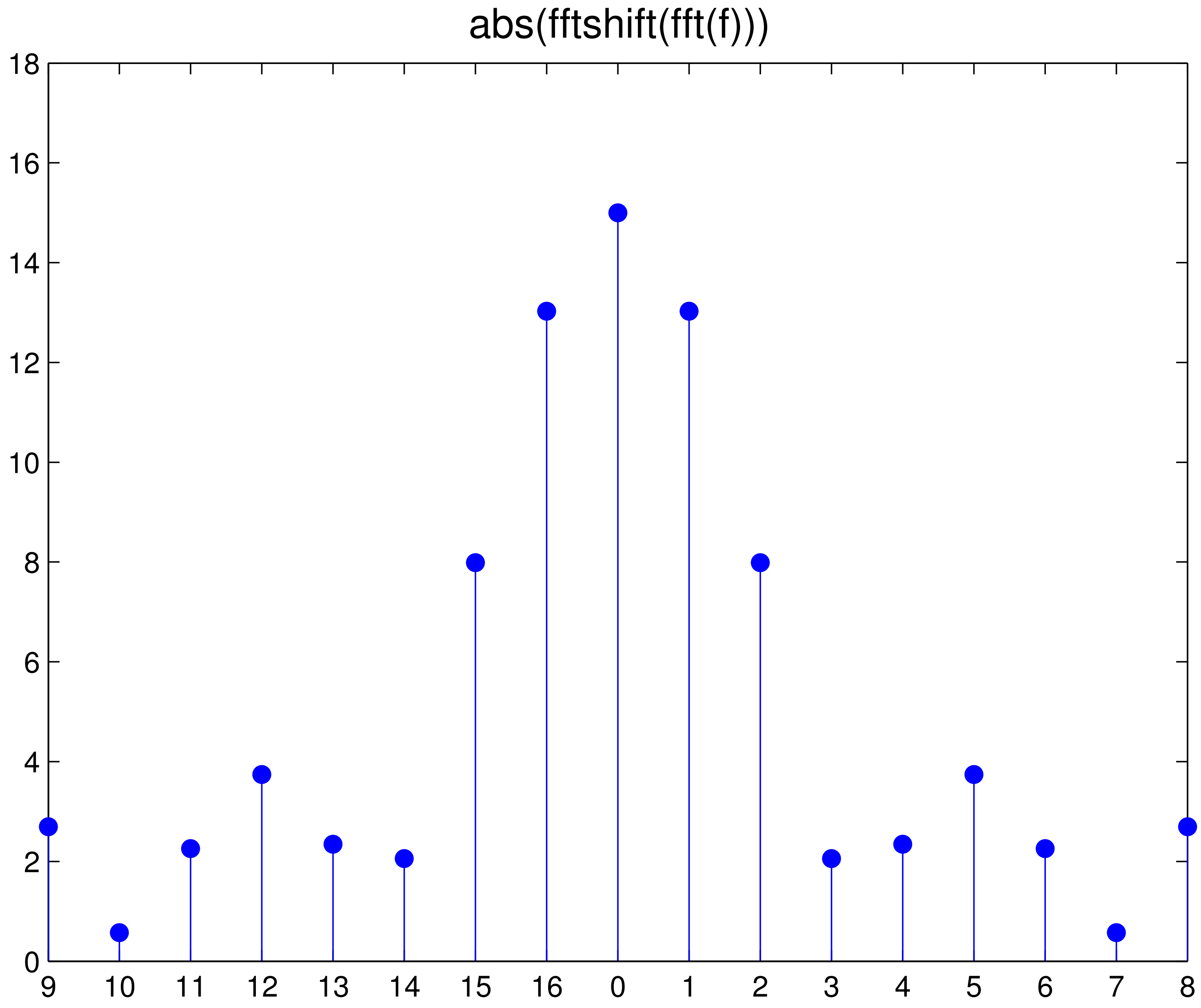
Conjugate symmetry in practice – Fourier image



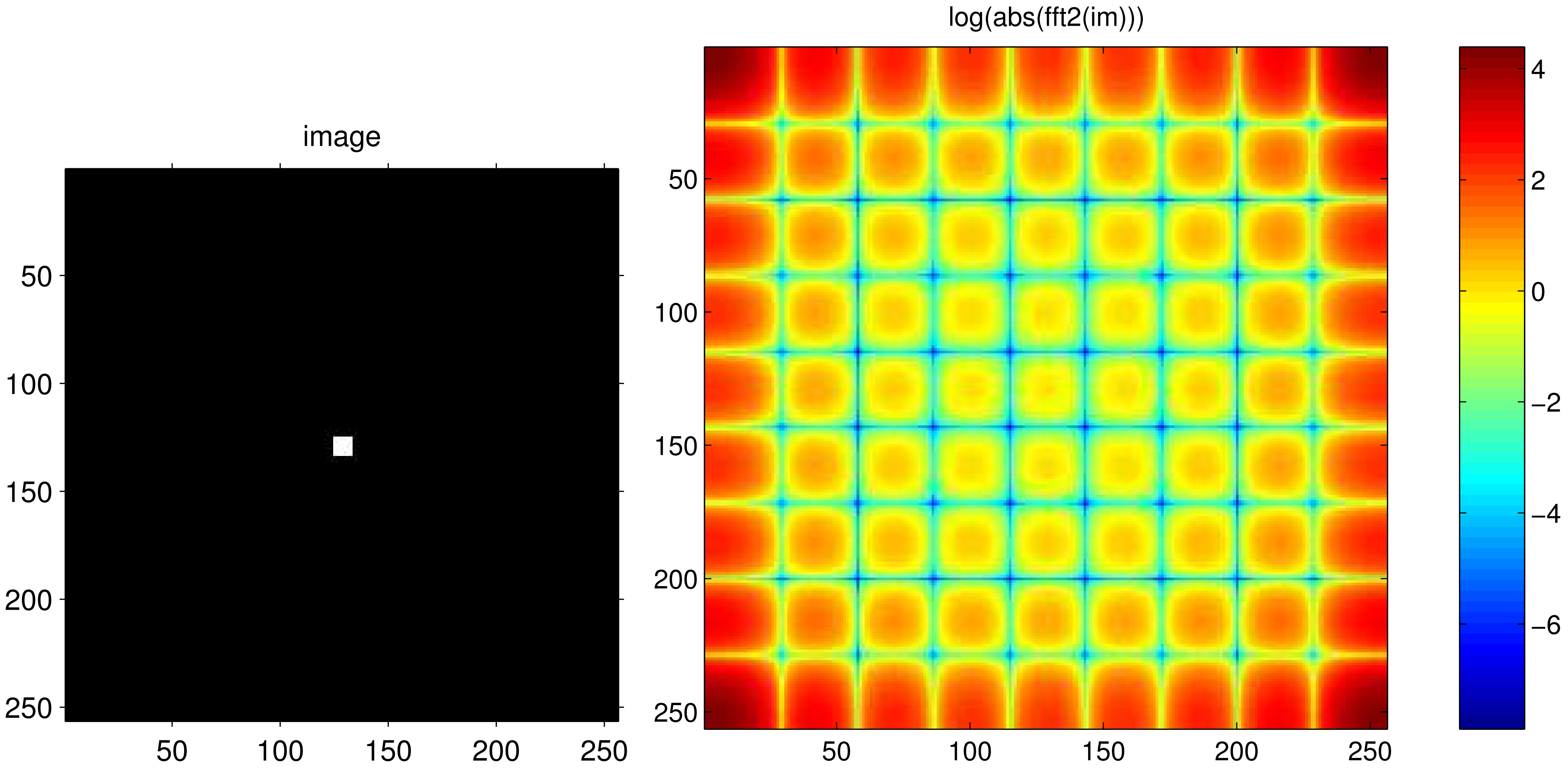


Conjugate symmetry in practice – F with centered origin

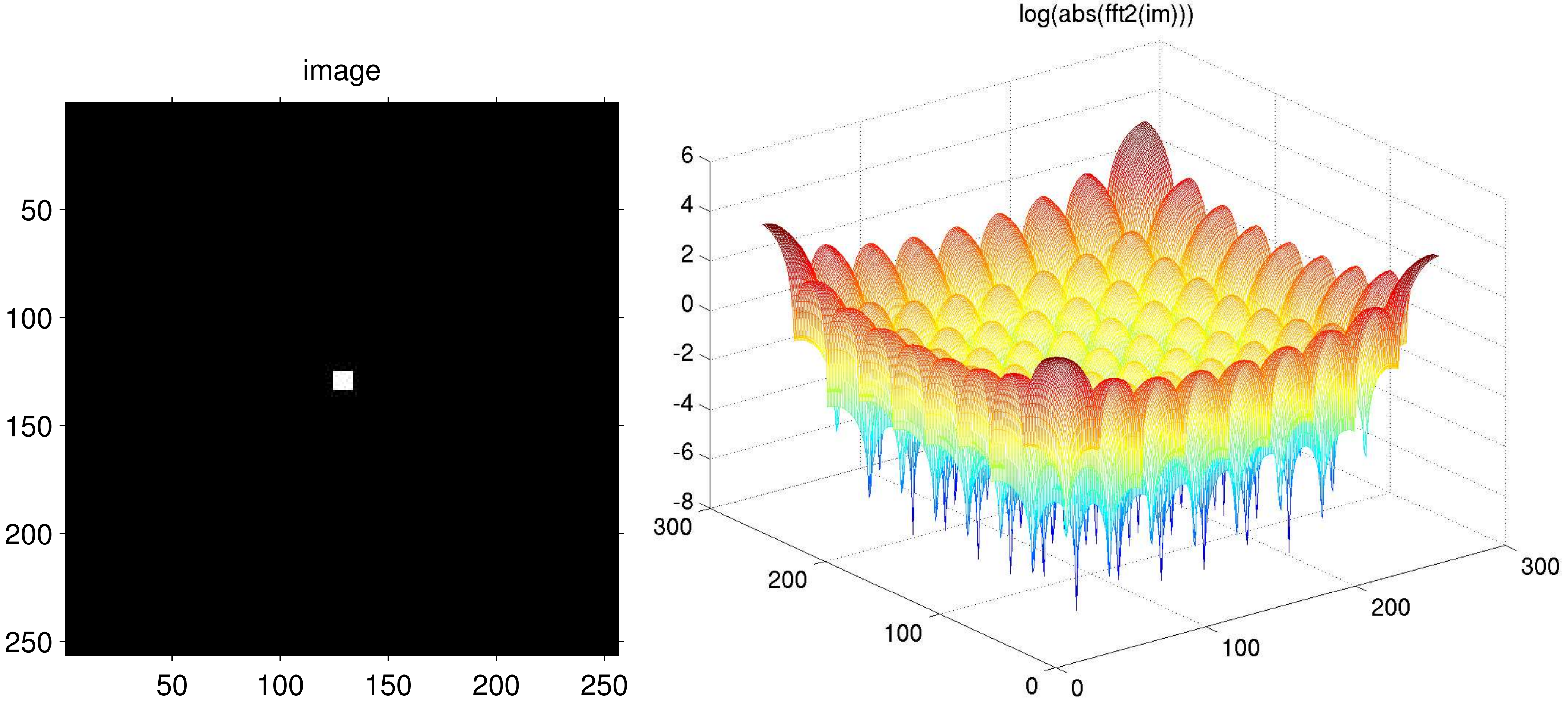




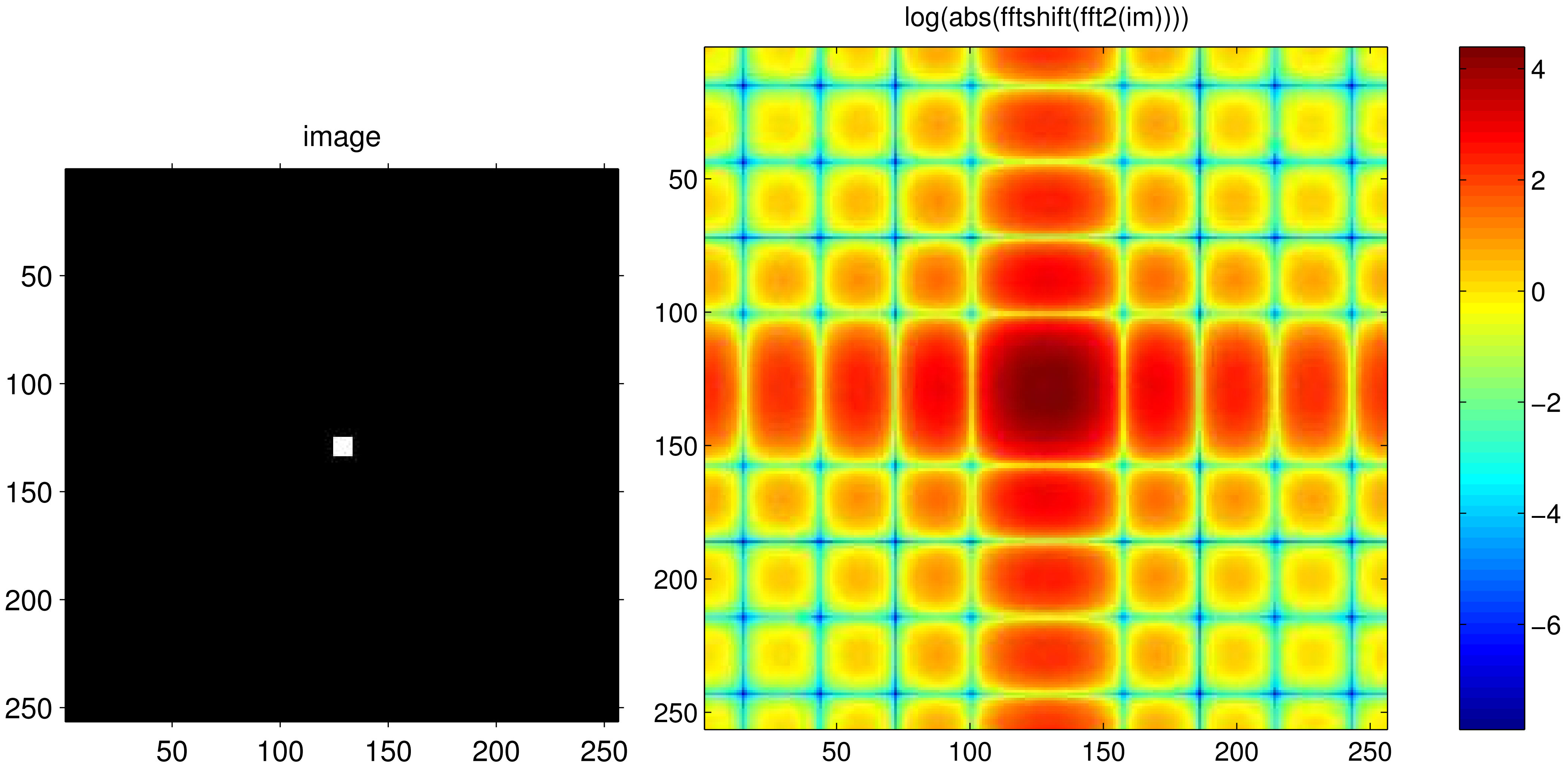
Conjugate symmetry in images



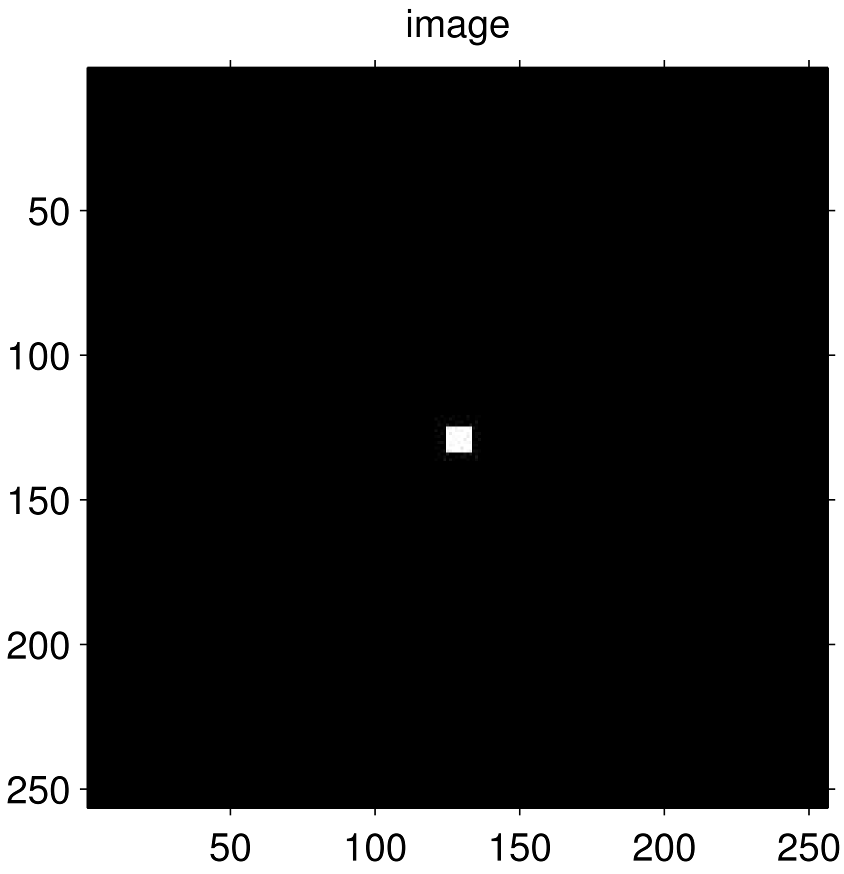
Conjugate symmetry in images

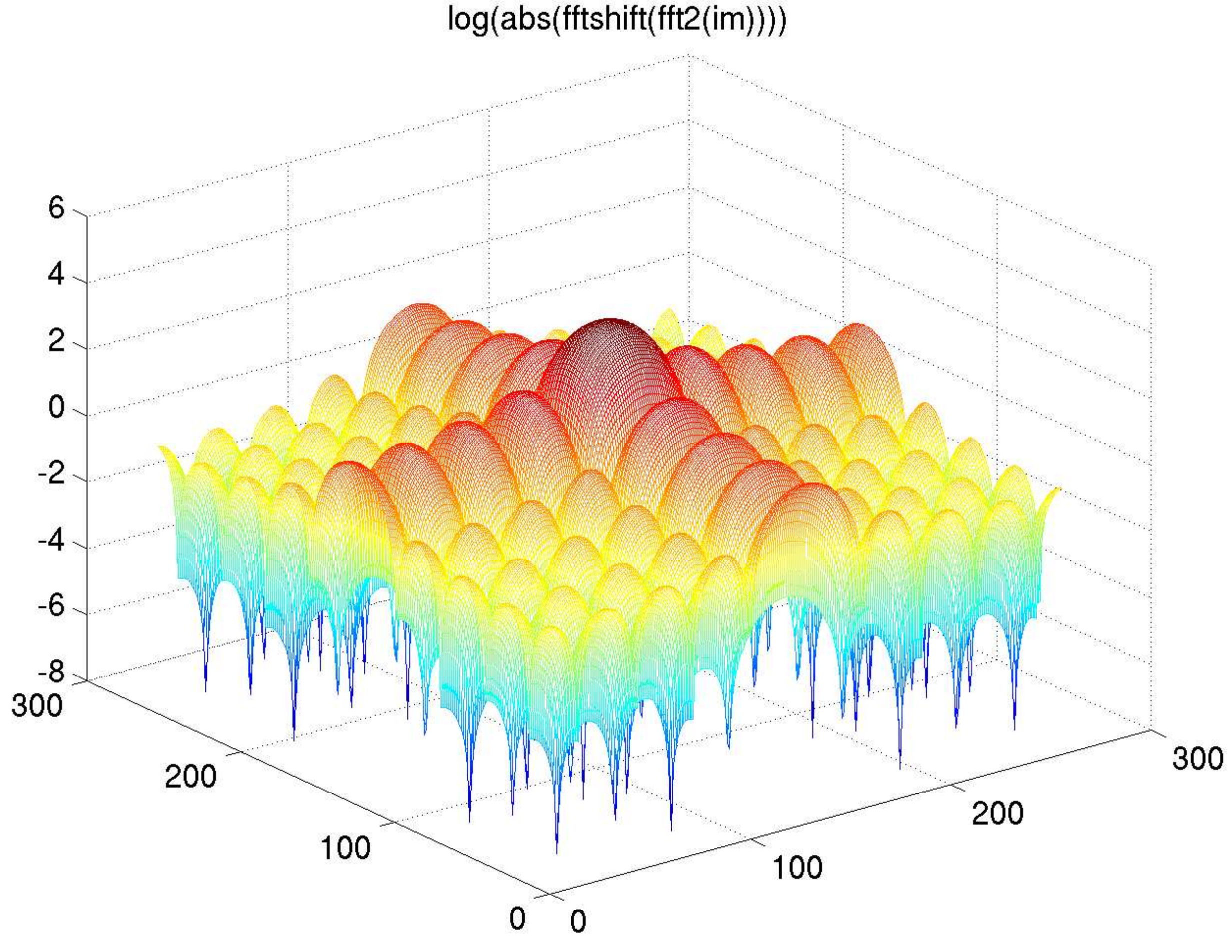


Conjugate symmetry in images – centered origin



Conjugate symmetry in images – centered origin





Fourier pair of rectangle function

$f_A(x) = A$ for $0 \leq x \leq X$ and $f_A(x) = 0$ elsewhere.

For the derivation we will need:

Fourier pair of rectangle function

$f_A(x) = A$ for $0 \leq x \leq X$ and $f_A(x) = 0$ elsewhere.

For the derivation we will need:

$$\sin(\omega x) = \frac{1}{2i}\left(e^{i\omega x} - e^{-i\omega x}\right)$$

Fourier pair of rectangle function

$f_A(x) = A$ for $0 \le x \le X$ and $f_A(x) = 0$ elsewhere.

For the derivation we will need:

$$\sin(\omega x) = \frac{1}{2i}\left(e^{i\omega x} - e^{-i\omega x}\right)$$

$$\mathrm{sinc}(\omega x) = \frac{\sin(\omega x)}{\omega x}$$

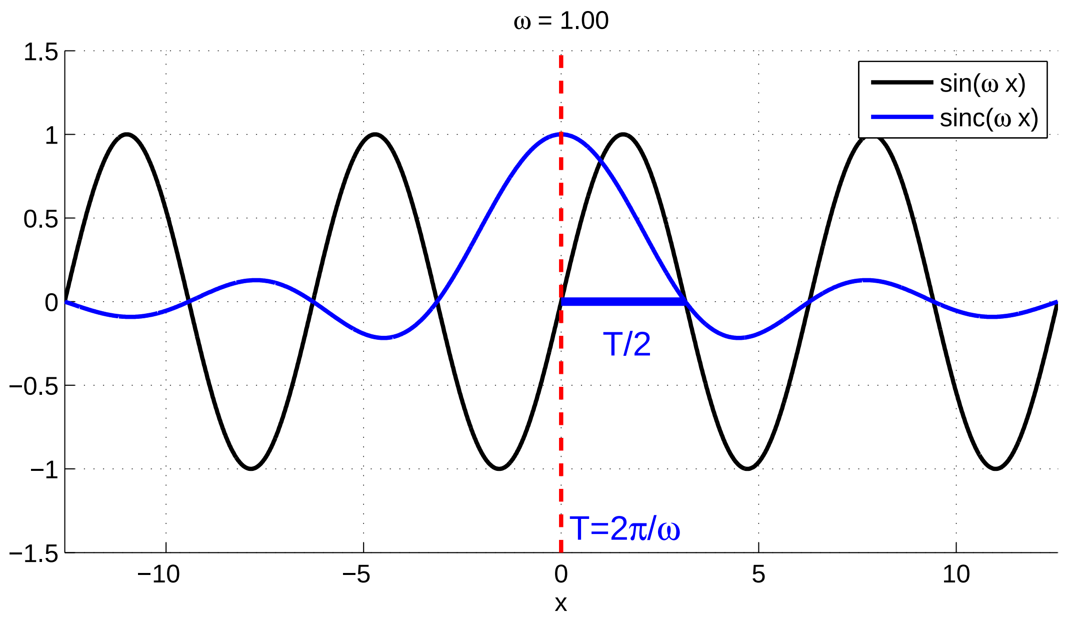
Fourier pair of rectangle function

$f_A(x) = A$ for $0 \leq x \leq X$ and $f_A(x) = 0$ elsewhere.

For the derivation we will need:

$$\sin(\omega x) = \frac{1}{2i}\left(e^{i\omega x} - e^{-i\omega x}\right)$$

$$\mathrm{sinc}(\omega x) = \frac{\sin(\omega x)}{\omega x}$$



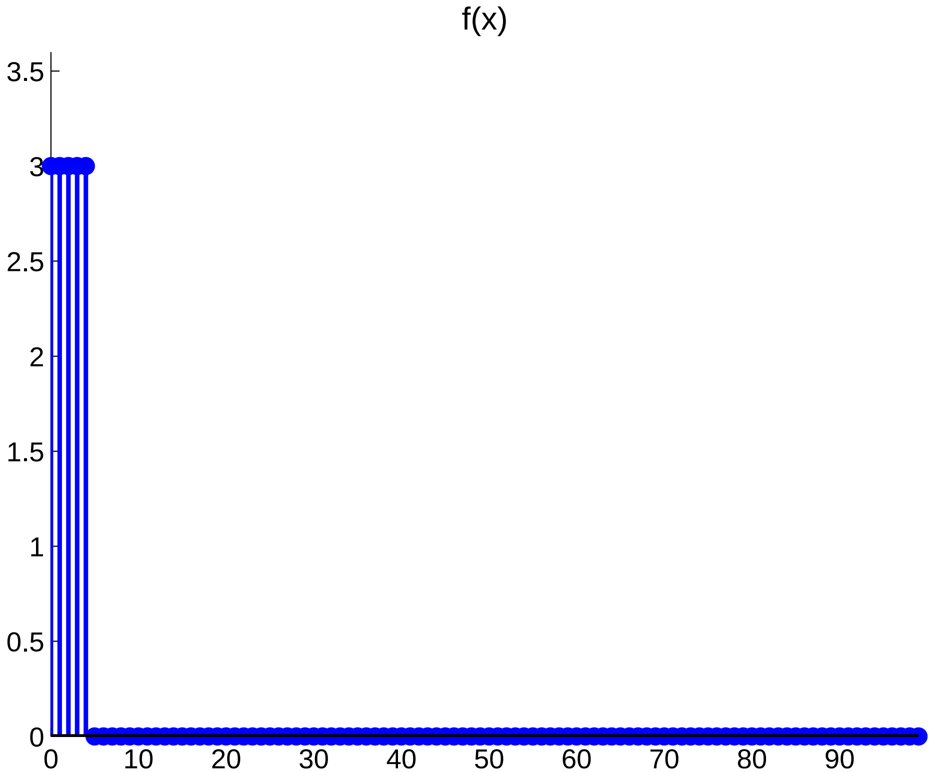
Fourier pair of rectangle function

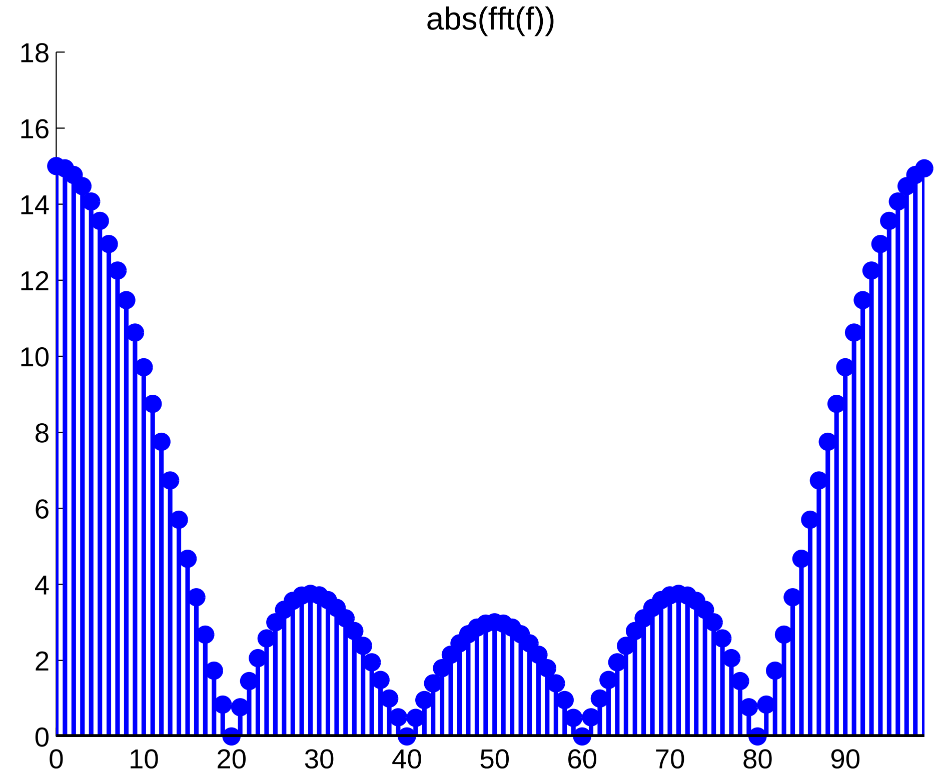
$$
\begin{aligned}
F(u) &= \int_{-\infty}^{\infty} f(x) e^{-i2\pi ux} dx \\
&= A \int_{0}^{X} e^{-i2\pi ux} dx \\
&= A \left[\frac{e^{-i2\pi ux}}{-i2\pi u} \right]_0^X \\
&= \frac{A}{-i2\pi u} \left(e^{-i2\pi Xu} - 1 \right) \\
&= \frac{A}{-i2\pi u} (-e^{-i\pi Xu}) \left(e^{i\pi Xu} - e^{-i\pi Xu} \right) \\
&= \frac{A}{\pi u} e^{-i\pi Xu} \left(\frac{e^{i\pi Xu} - e^{-i\pi Xu}}{2i} \right) \\
&= \frac{A}{\pi u} e^{-i\pi Xu} \sin \pi Xu \\
&= AXe^{-i\pi Xu} \frac{\sin \pi Xu}{\pi Xu} \\
&= AXe^{-i\pi Xu} \mathrm{sinc}(\pi Xu)
\end{aligned}
$$

which implies

$$|F(u)| = AX|\mathrm{sinc}(\pi Xu)|$$

Fourier pair of discrete rectangle function





$A = 3, X = 5, M = 100$ makes $T/2$ of sinc equal to 20.

$\mathcal{F}\{f_A(x) = A \text{ for } 0 \geq x \leq X\} = AXe^{-i\pi Xu}\mathrm{sinc}(\frac{\pi X}{M}u)$

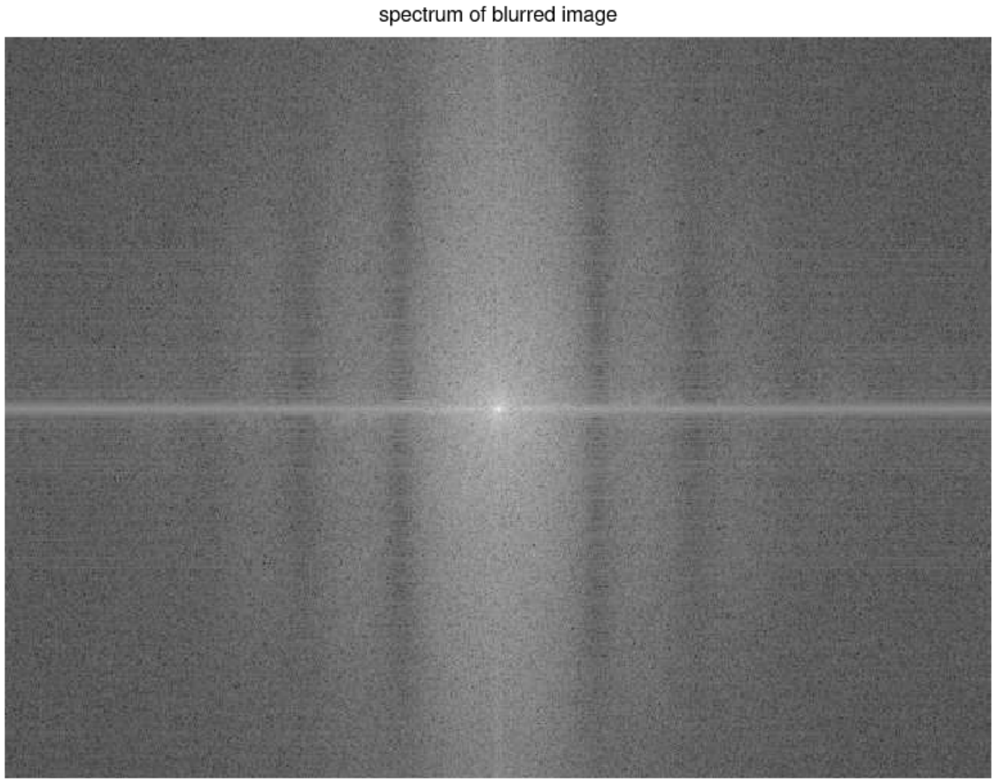
Remeber that $T = 2\pi/\omega$, hence $X = \frac{2M}{T}$

We can find the frequency of $|F(u)|$ perhaps even simpler . . . [1]

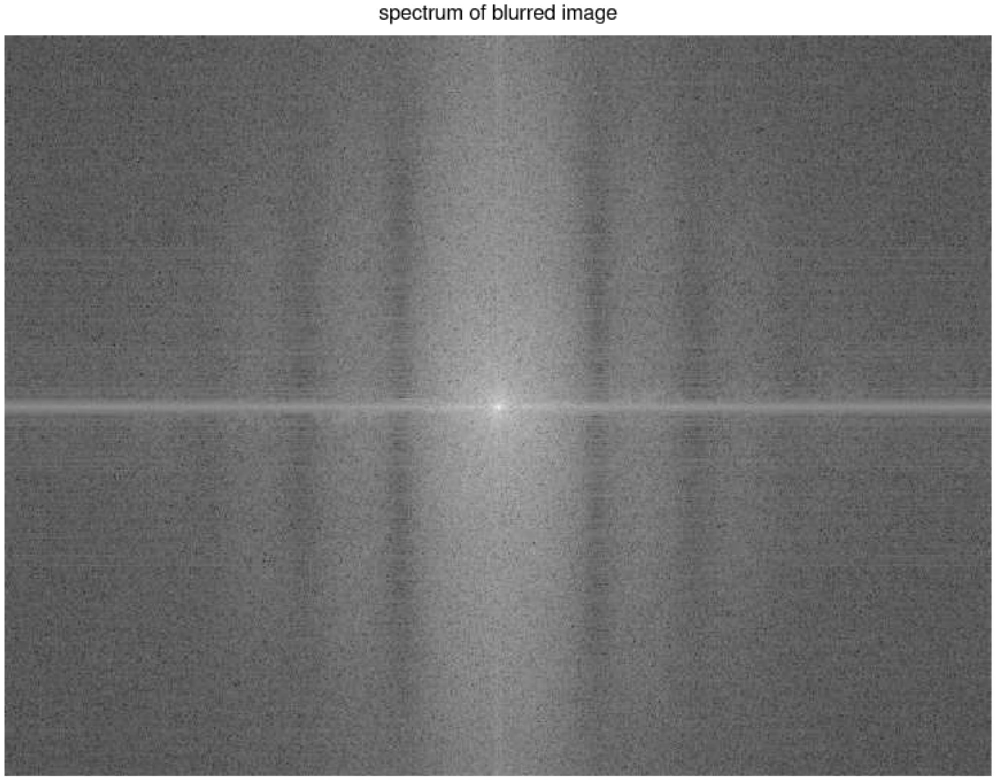
[1] Blackboard only for the moment.

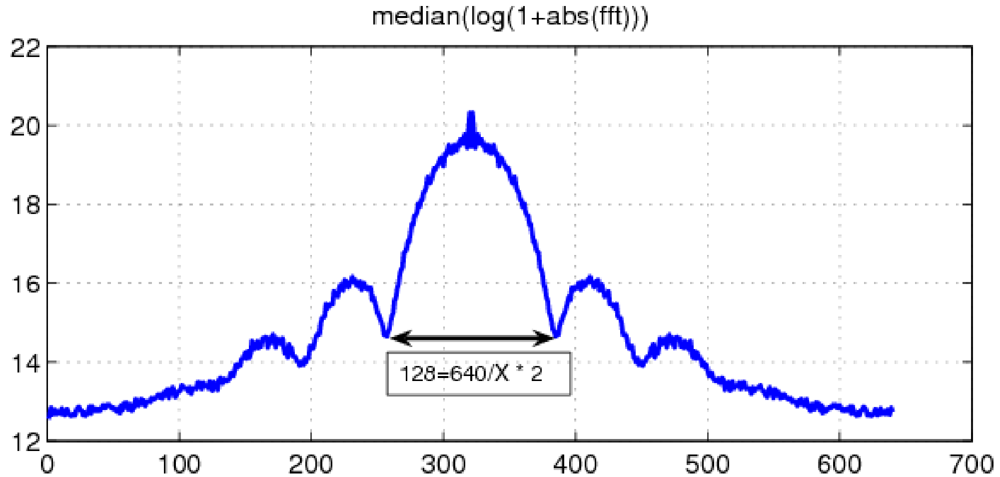
Use of Rectangle function — Image Restoration

Use of Rectangle function — Image Restoration



Use of Rectangle function — Image Restoration





Use of Rectangle function — Image Restoration

Blurred image.

Use of Rectangle function — Image Restoration

Blurred image.

Restored image.

Image processing in frequency domain — Filtering

Image processing in frequency domain — Filtering

1. $F(u,v) = \mathcal{F}\{f(x,y)\}$

Image processing in frequency domain — Filtering

1. $F(u,v) = \mathcal{F}\{f(x,y)\}$
2. $G(u,v) = H(u,v) .* F(u,v)$, where $.*$ means "per element" multiplication.

Image processing in frequency domain — Filtering

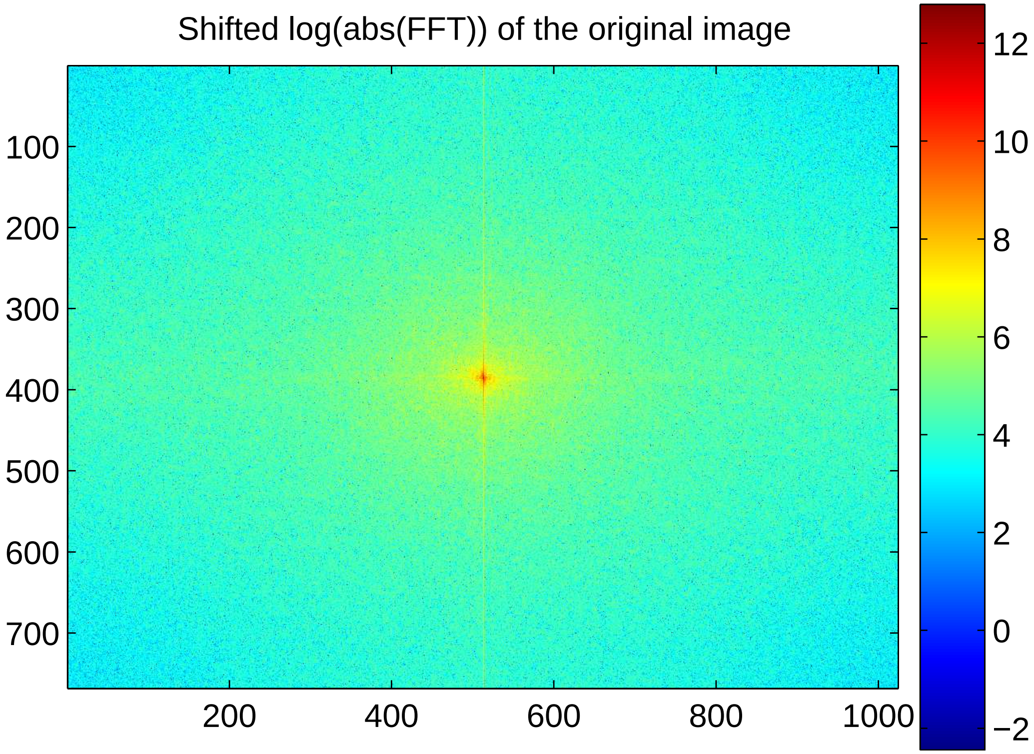
1. $F(u,v) = \mathcal{F}\{f(x,y)\}$
2. $G(u,v) = H(u,v). * F(u,v)$, where $.*$ means “per element” multiplication.
3. $g(x,y) = \mathcal{F}^{-1}\{G(u,v)\}$

Image processing in frequency domain — Filtering

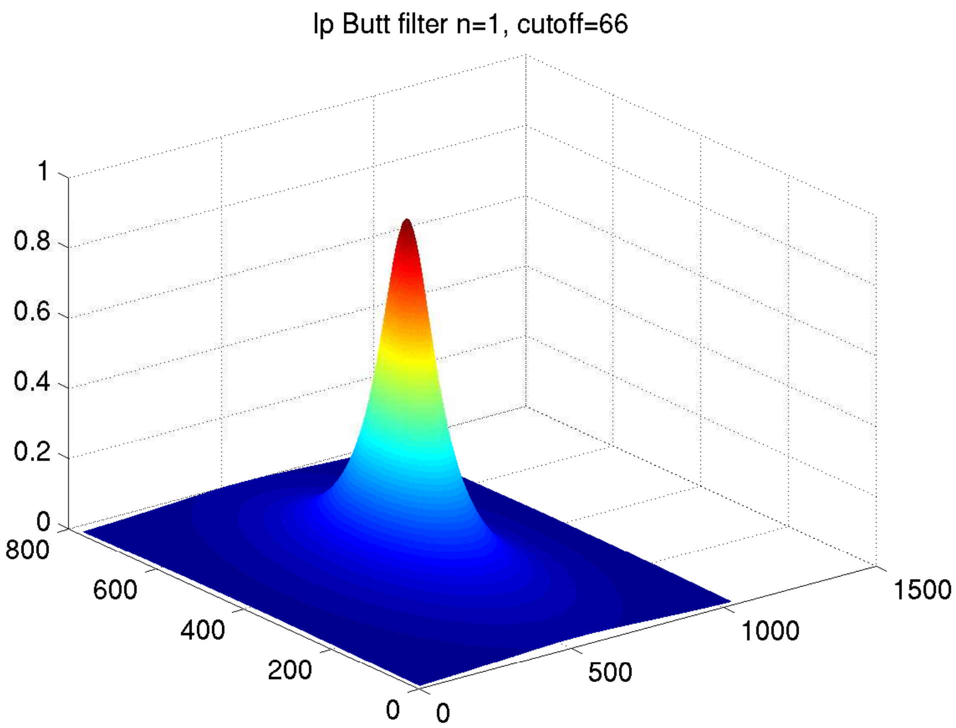
1. $F(u,v) = \mathcal{F}\{f(x,y)\}$
2. $G(u,v) = H(u,v). * F(u,v)$, where $.*$ means "per element" multiplication.
3. $g(x,y) = \mathcal{F}^{-1}\{G(u,v)\}$

Do not forget: We display $\ln|F(u,v)|$. The filter must be applied to the $F(u,v)$.

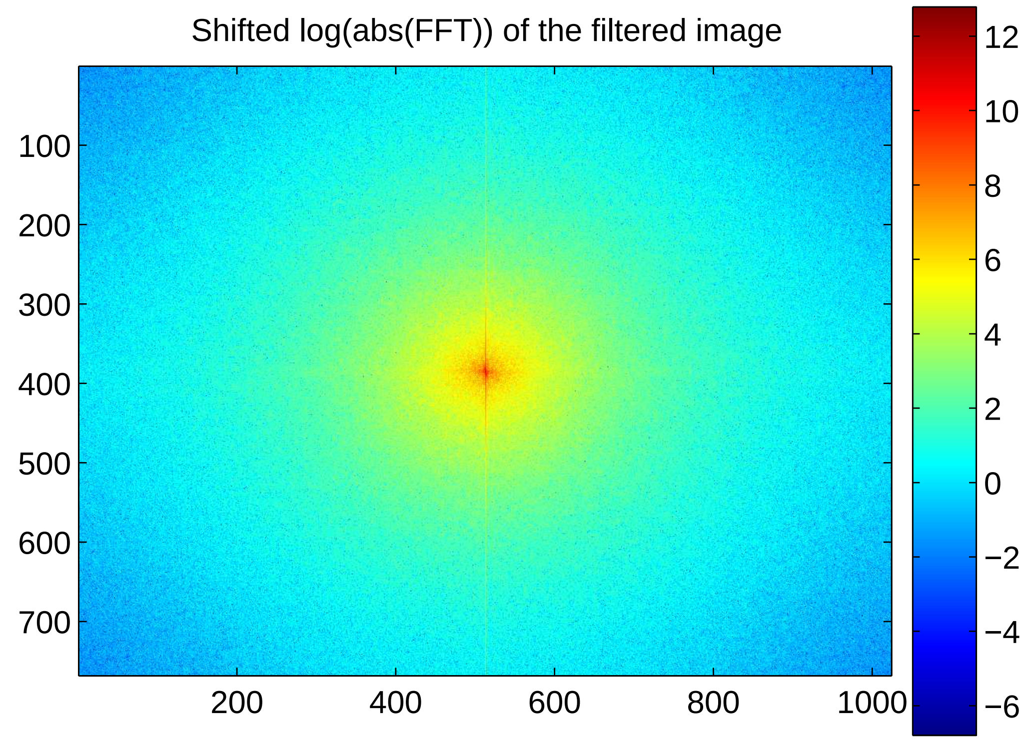
Lowpass filtering — Butterworth filter I



Lowpass filtering — Butterworth filter II



Butterworth lowpass filter



FFT of the filtered image

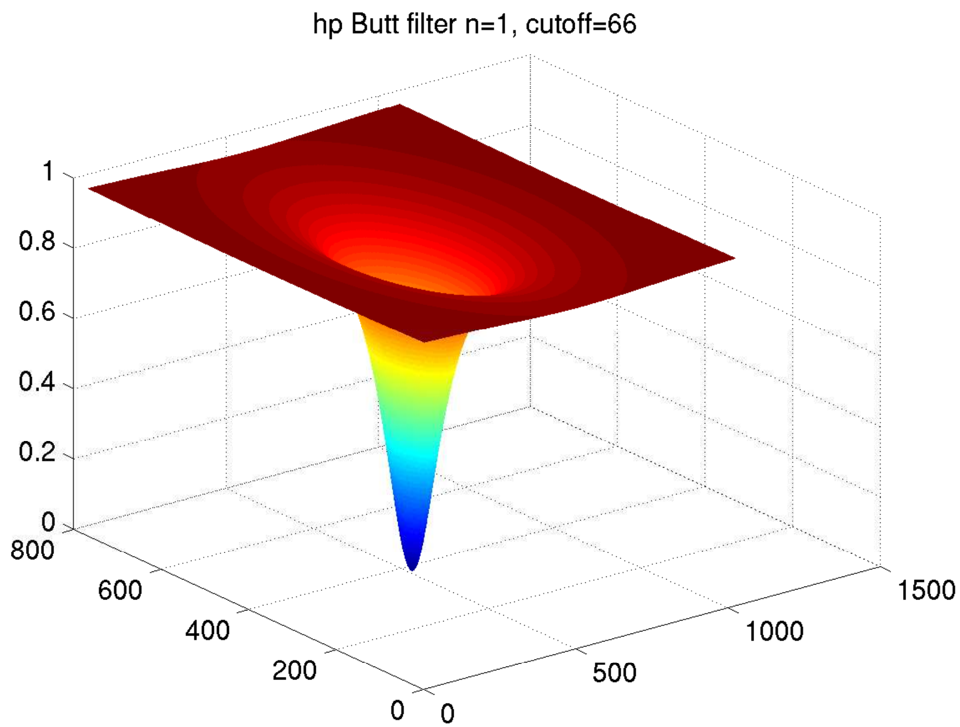
$H_{lp}(u,v) = \frac{1}{1+(D(u,v)/D_0)^{2/n}}$, where $D(u,v) = \sqrt{u^2+v^2}$

Lowpass filtering — Butterworth filter III

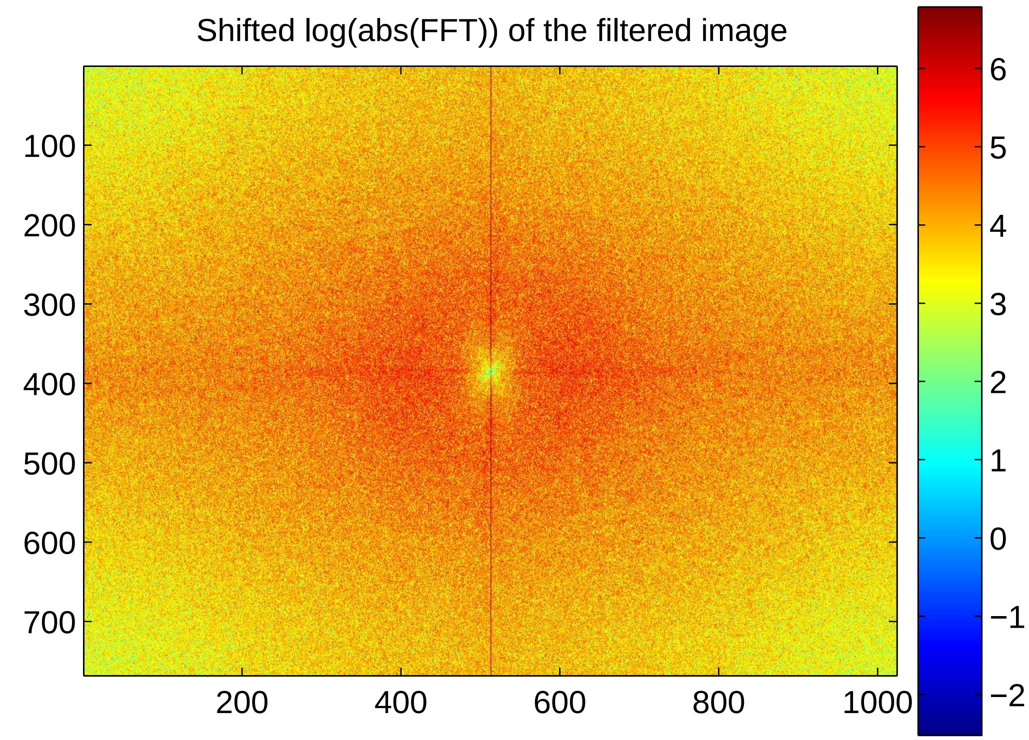
Original image

Filtered image

Highpass filtering — Butterworth filter I



Butterworth highpass filter



FFT of the filtered image

$$H_{hp}(u,v) = 1 - H_{lp}(u,v)$$

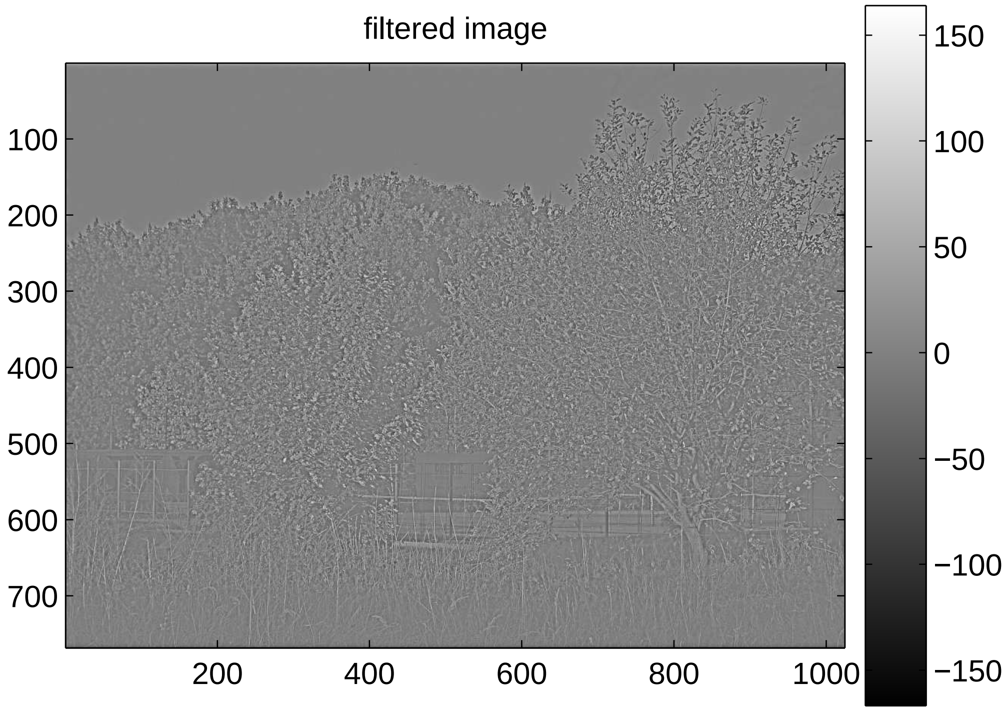
Highpass filtering — Butterworth filter II

Original image

Filtered image

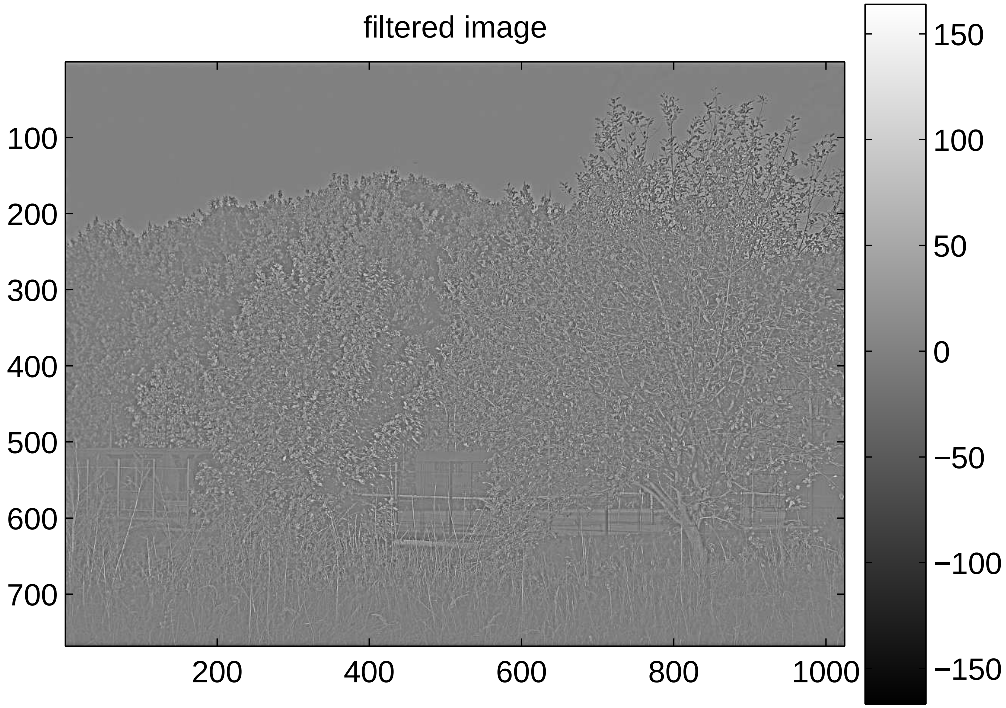
Why do we not see anything in the filtered image?

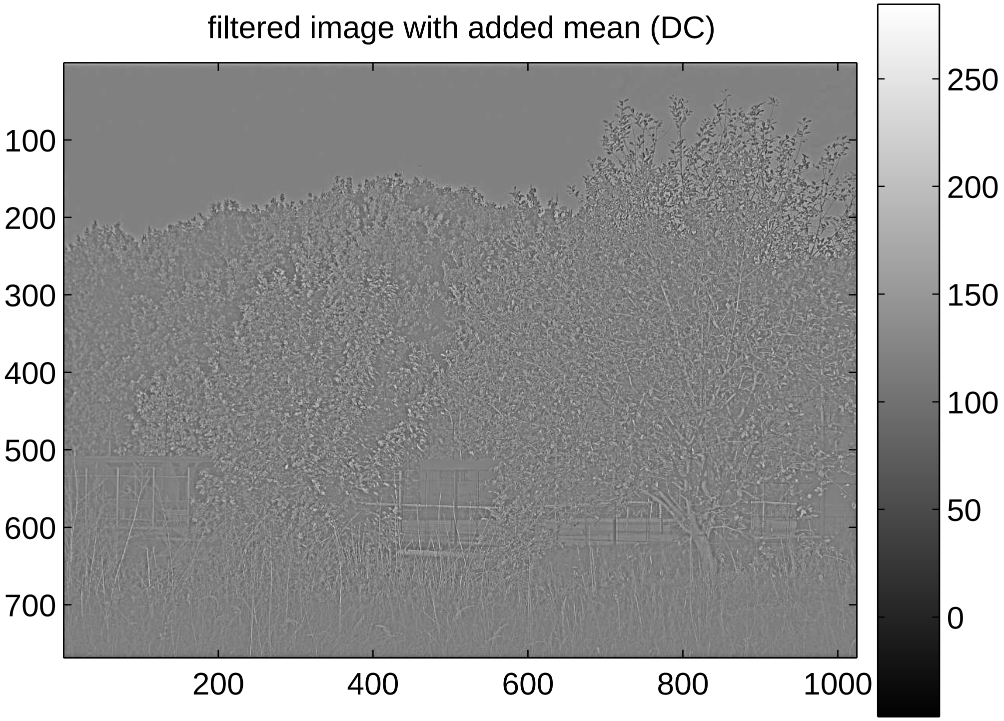
Highpass filtering — cont.



Some values are negative. Why?

Highpass filtering — cont.

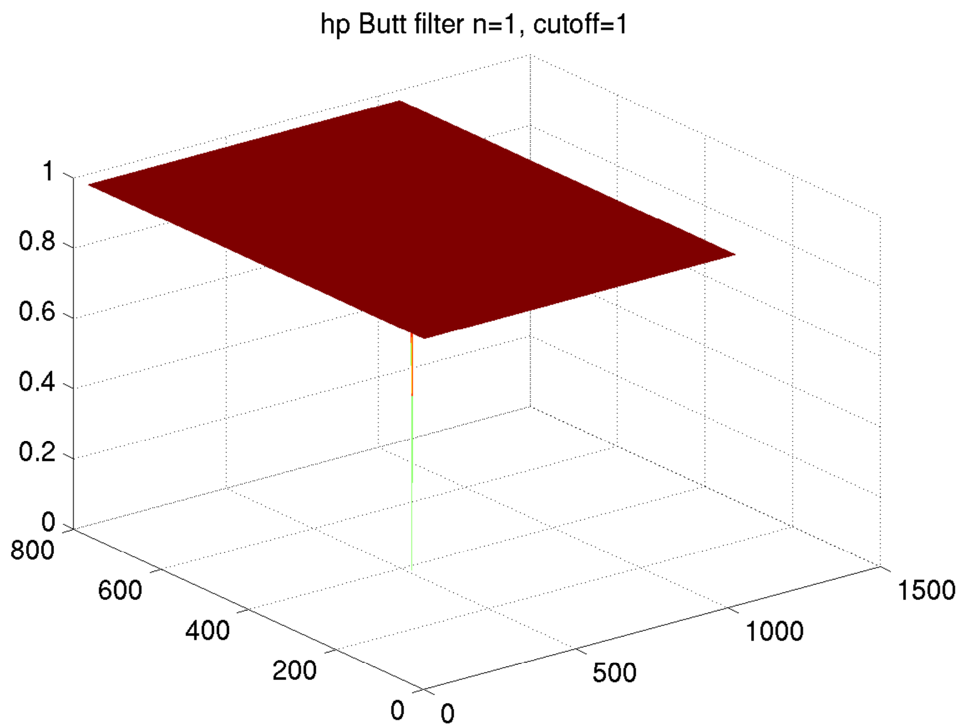




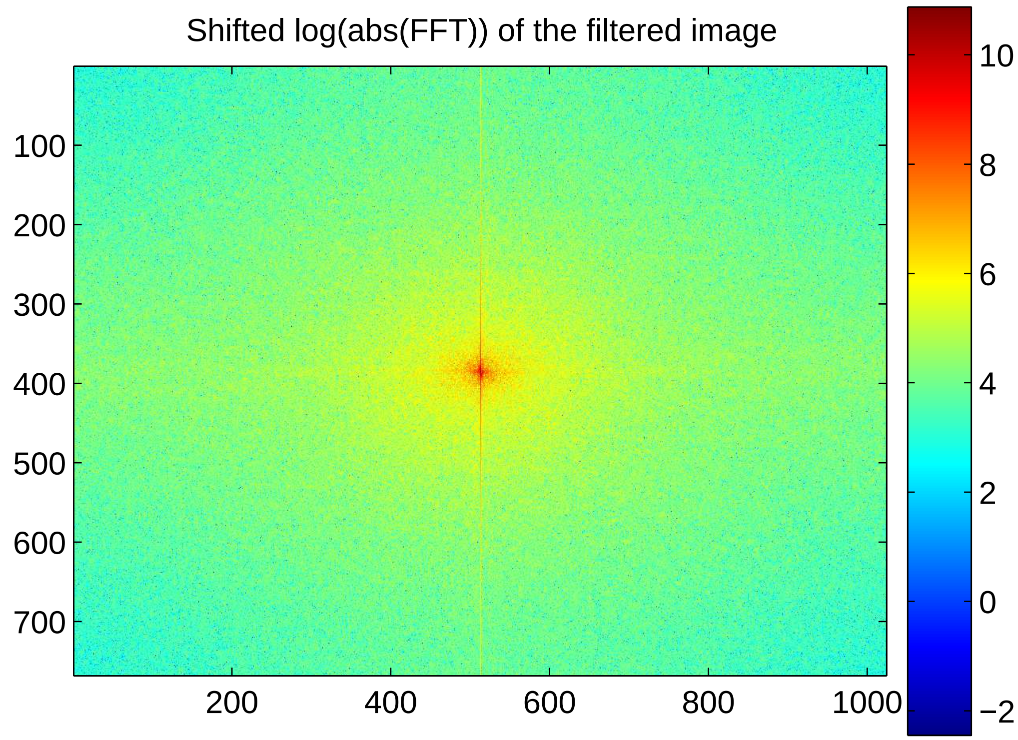
Some values are negative. Why?

We lost the DC part of the FFT!

Highpass filtering — Loosing DC part I



Butterworth highpass filter



FFT of the filtered image

$$H_{hp}(u,v) = 1 - H_{lp}(u,v)$$

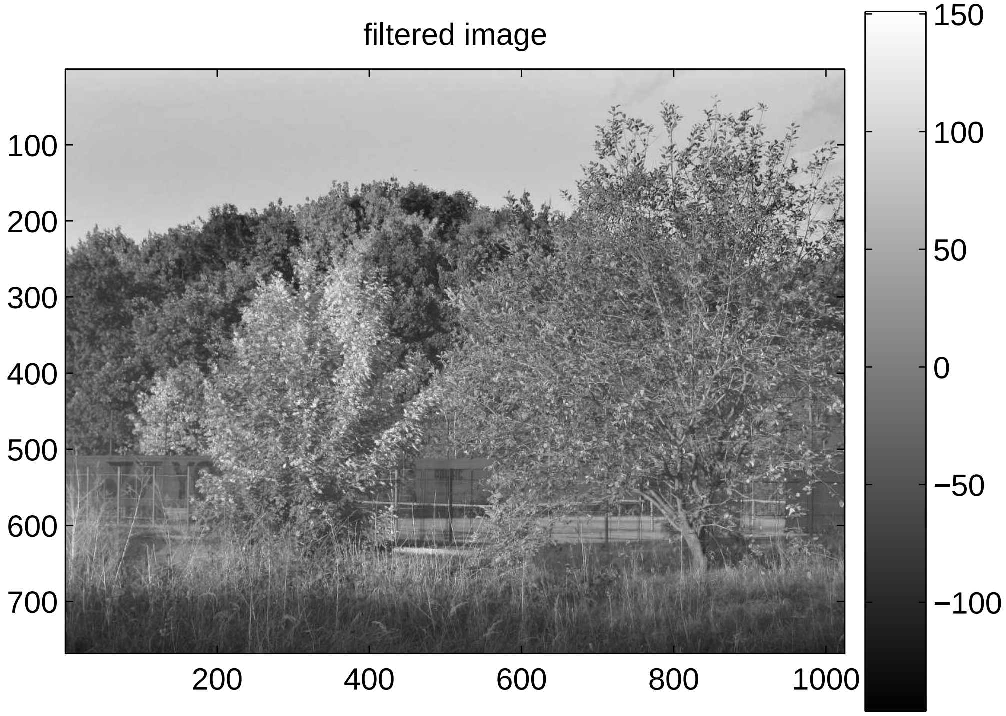
Highpass filtering — Loosing DC part II

Original image

Filtered image

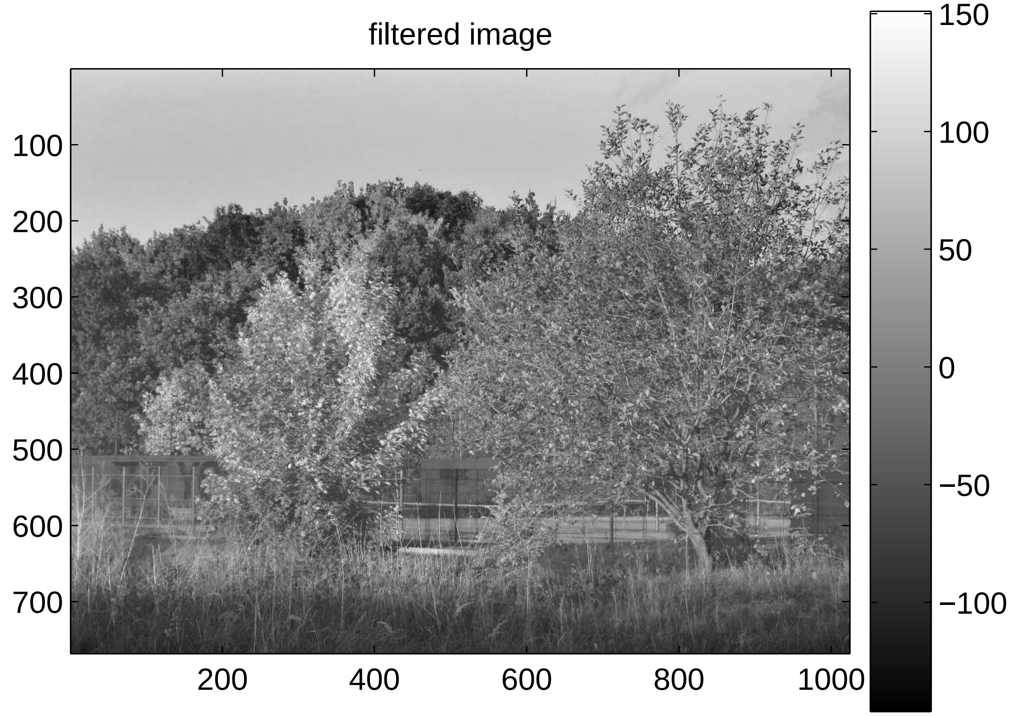
Despite a very gentle high-pass filter the filtered images does not resemble the original one.

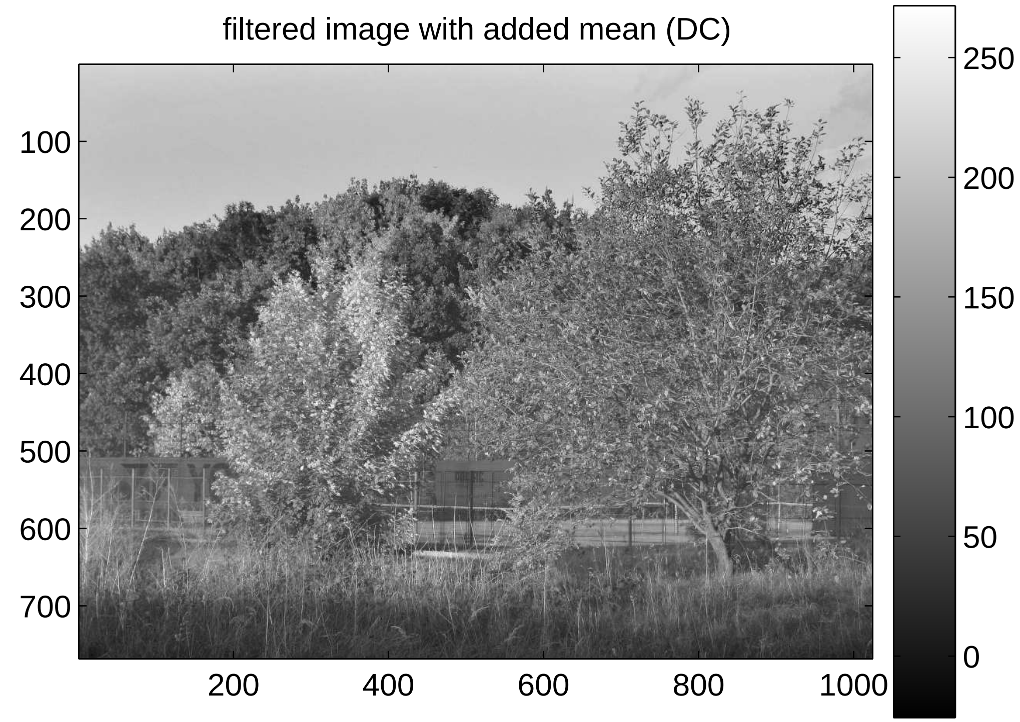
Highpass filtering — Loosing DC part III



Some values are negative.

Highpass filtering — Loosing DC part III





Some values are negative.

Adding a DC part (mean of the original image) would correct the image.

More advanced filtering — Homomorphic filtering

Idea: simultaneously normalize the brightness across an image and increase contrast.

More advanced filtering — Homomorphic filtering

Idea: simultaneously normalize the brightness across an image and increase contrast.

Image is a product of illumination and reflectance components:
$f(x,y) = i(x,y)r(x,y)$

More advanced filtering — Homomorphic filtering

Idea: simultaneously normalize the brightness across an image and increase contrast.

Image is a product of illumination and reflectance components:
$f(x, y) = i(x, y)r(x, y)$

Illumination i — slow spatial variations (low frequency)

More advanced filtering — Homomorphic filtering

Idea: simultaneously normalize the brightness across an image and increase contrast.

Image is a product of illumination and reflectance components:
$f(x, y) = i(x, y)r(x, y)$

Illumination i — slow spatial variations (low frequency)

Reflectance r — fast varitations (dissimilar objects)

More advanced filtering — Homomorphic filtering

Idea: simultaneously normalize the brightness across an image and increase contrast.

Image is a product of illumination and reflectance components:
$f(x,y) = i(x,y)r(x,y)$

Illumination i — slow spatial variations (low frequency)

Reflectance r — fast varitations (dissimilar objects)

Use logarithm to separate the components and filter the logarithms!

Homomorphic filtering — cont.

$$z(x,y) = \ln f(x,y) = \ln i(x,y) + \ln r(x,y)$$

Homomorphic filtering — cont.

$$z(x,y) = \ln f(x,y) = \ln i(x,y) + \ln r(x,y)$$

Fourier pair

$$Z(u,v) = I(u,v) + R(u,v)$$

Homomorphic filtering — cont.

$$z(x,y) = \ln f(x,y) = \ln i(x,y) + \ln r(x,y)$$

Fourier pair

$$Z(u,v) = I(u,v) + R(u,v)$$

Filtering

$$S(u,v) = H(u,v)Z(u,v) = H(u,v)I(u,v) + H(u,v)R(u,v)$$

Homomorphic filtering — cont.

$$z(x, y) = \ln f(x, y) = \ln i(x, y) + \ln r(x, y)$$

Fourier pair

$$Z(u, v) = I(u, v) + R(u, v)$$

Filtering

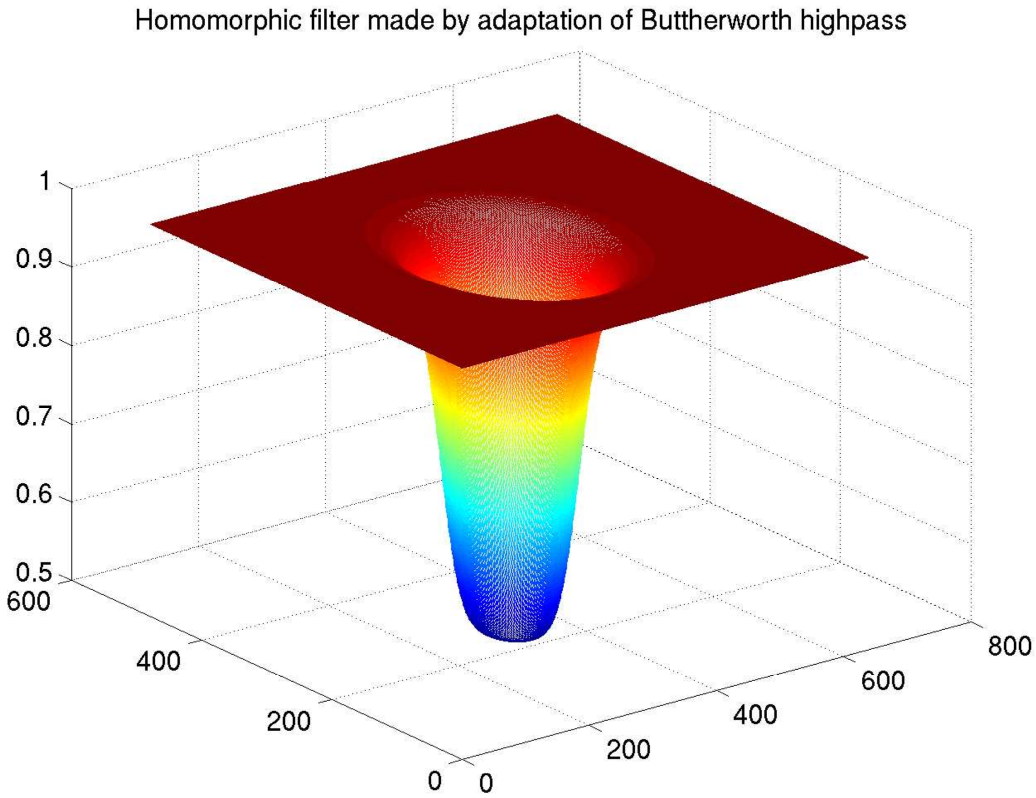
$$S(u, v) = H(u, v)Z(u, v) = H(u, v)I(u, v) + H(u, v)R(u, v)$$

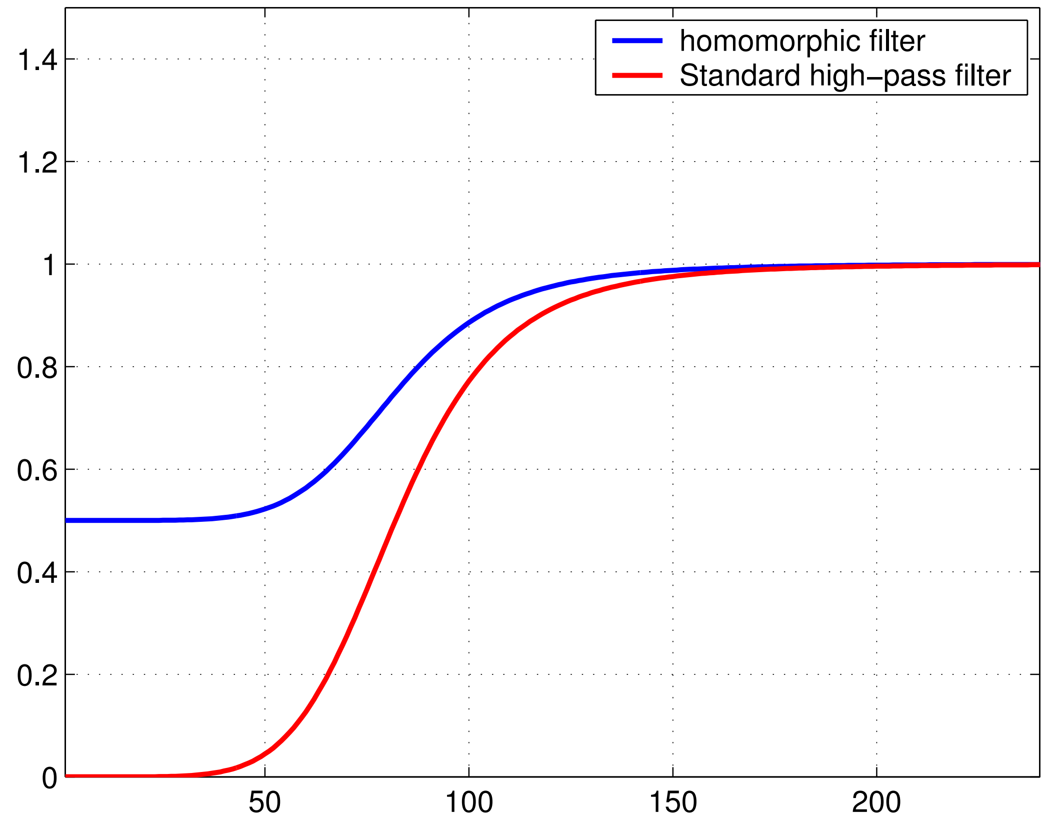
back to space $s(x, y) = \mathcal{F}^{-1}\{S(u, v)\}$ and back from $\ln$

$$g(x, y) = \exp\left(s(x, y)\right)$$

So, we can suppress variations in illumination and enhance reflectance component.

Homomorphic filtering — filters





Remember: The filter is applied to $Z(u, v)$. Not to $F(u, v)$!

Homomorphic filtering — results

Original image.

Homomorphic filtering — results

Original image.

Filtered image.

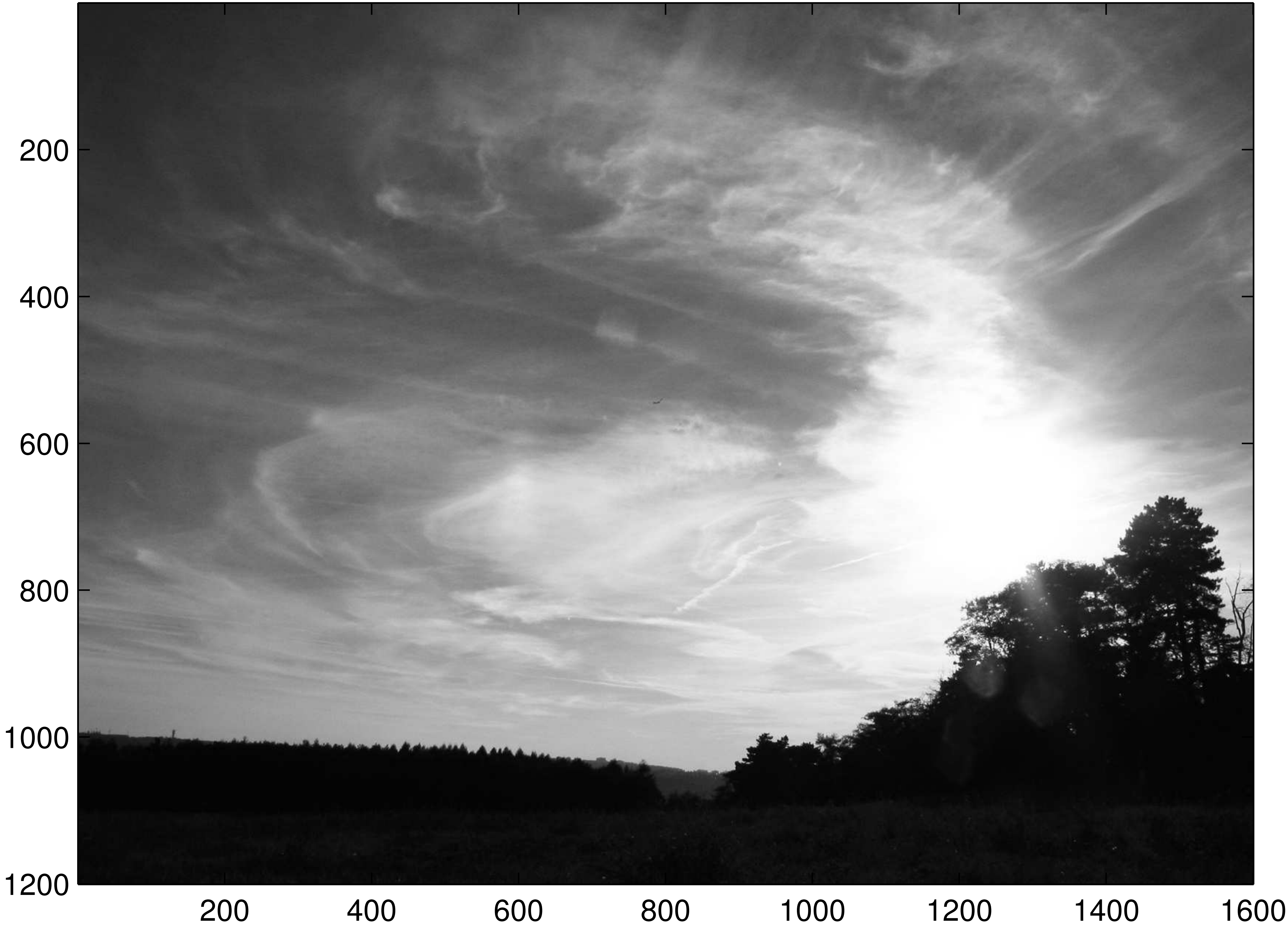
Readings

Many images and demos were made by using the codes from chapter 3 of the book [2]. Slighlty non-traditional and very valuable insight into Fourier imaging brings the book [1]

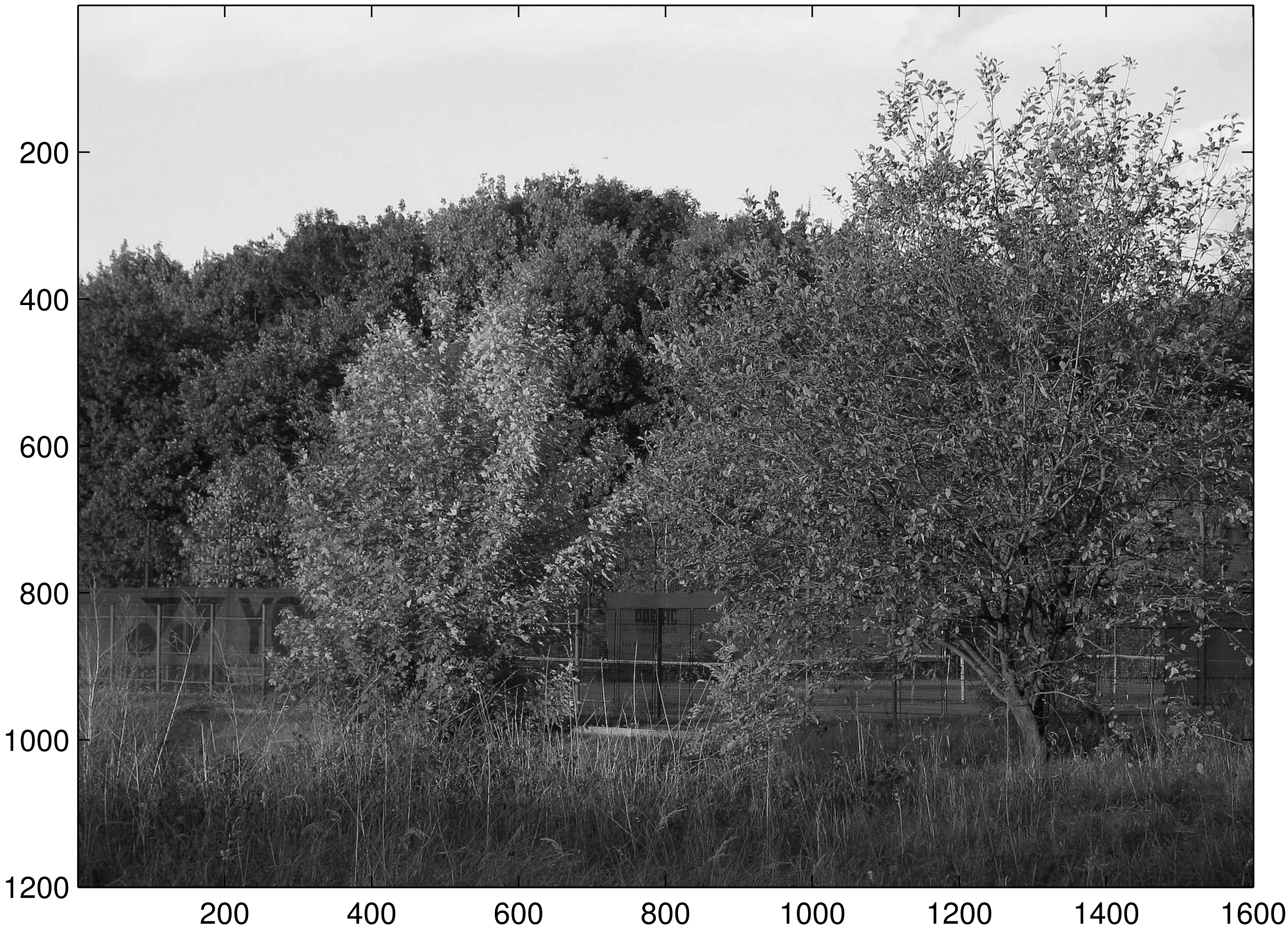
[1] Ronald N. Bracewell. Fourier analysis and imaging. Kluwer Academic/Plenum Publishers, New York, USA, 2003.

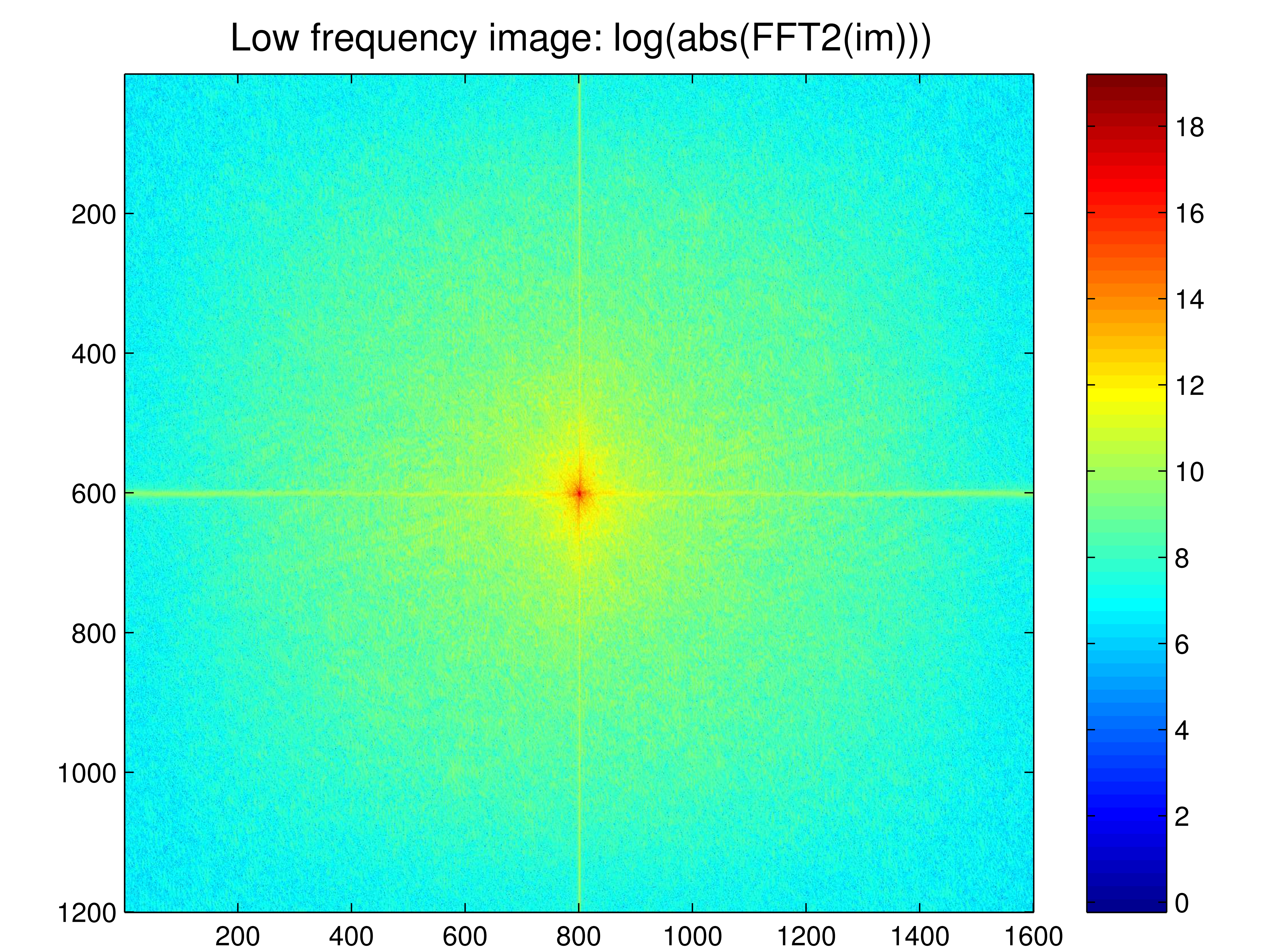
[2] Tomáš Svoboda, Jan Kybic, and Václav Hlaváč. Image Processing, Analysis and Machine Vision. A MATLAB Companion. Thomson, 2007. Accompanying www site `http://visionbook.felk.cvut.cz`.

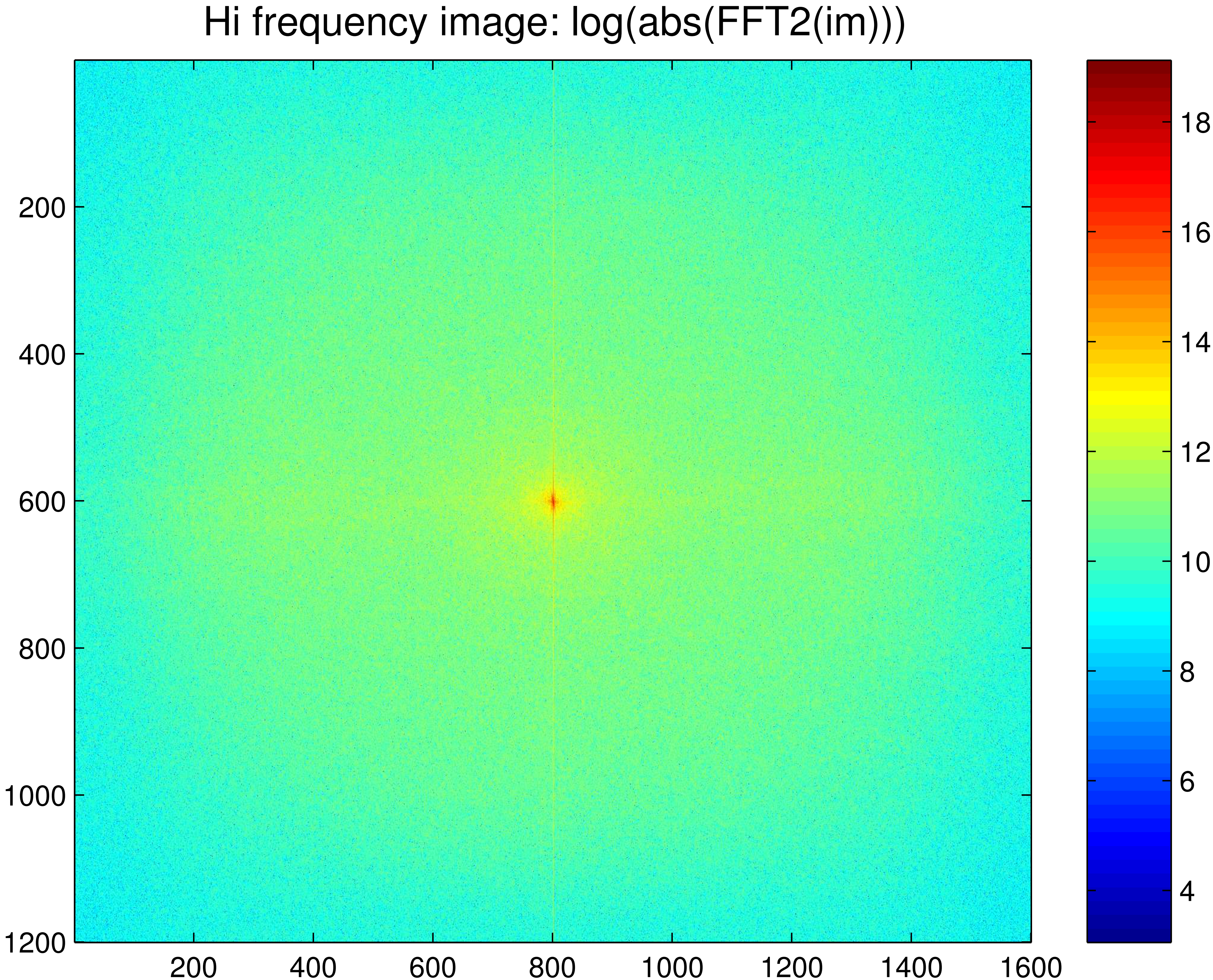
Low frequency image in gray scales

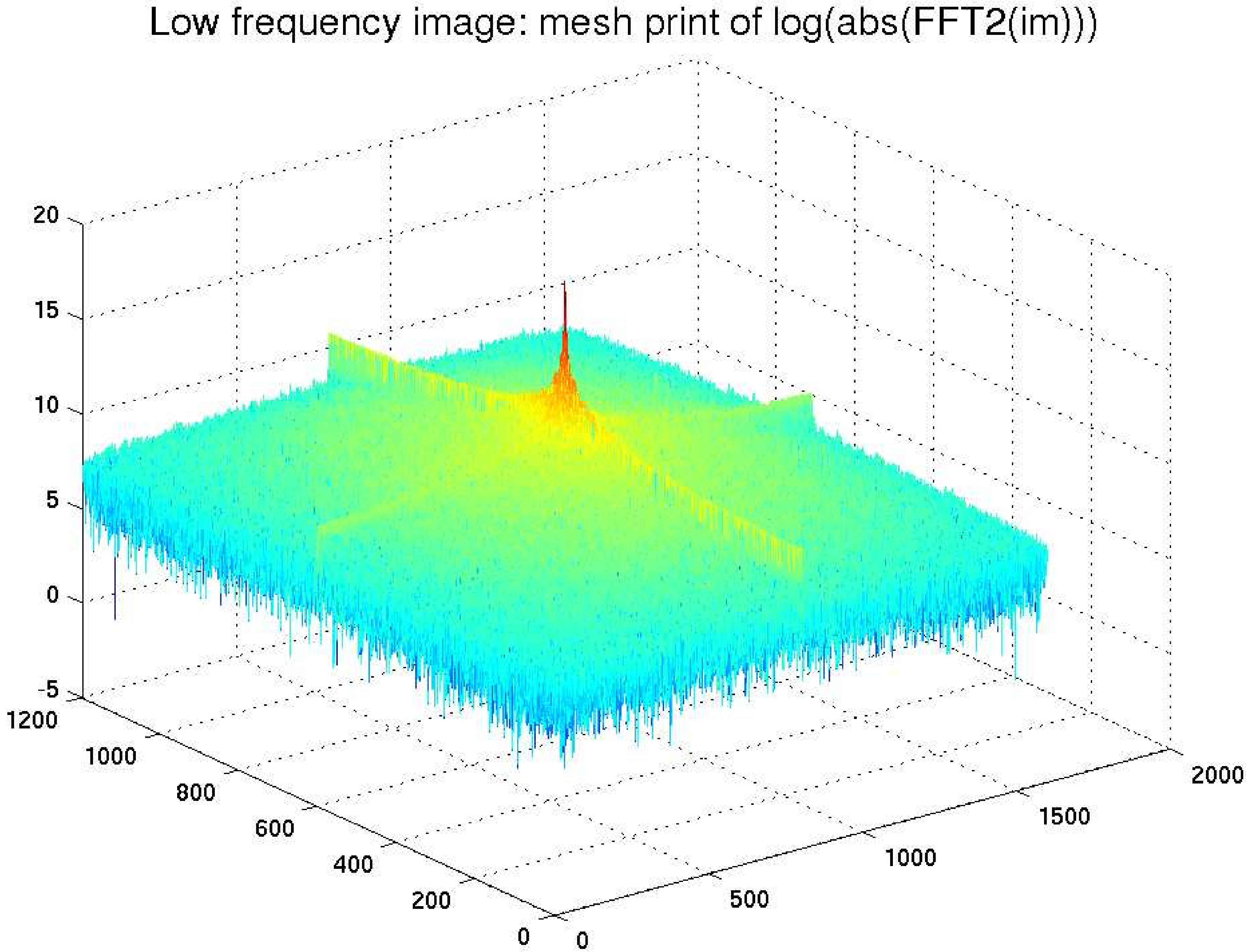


Hi frequency image in gray scales

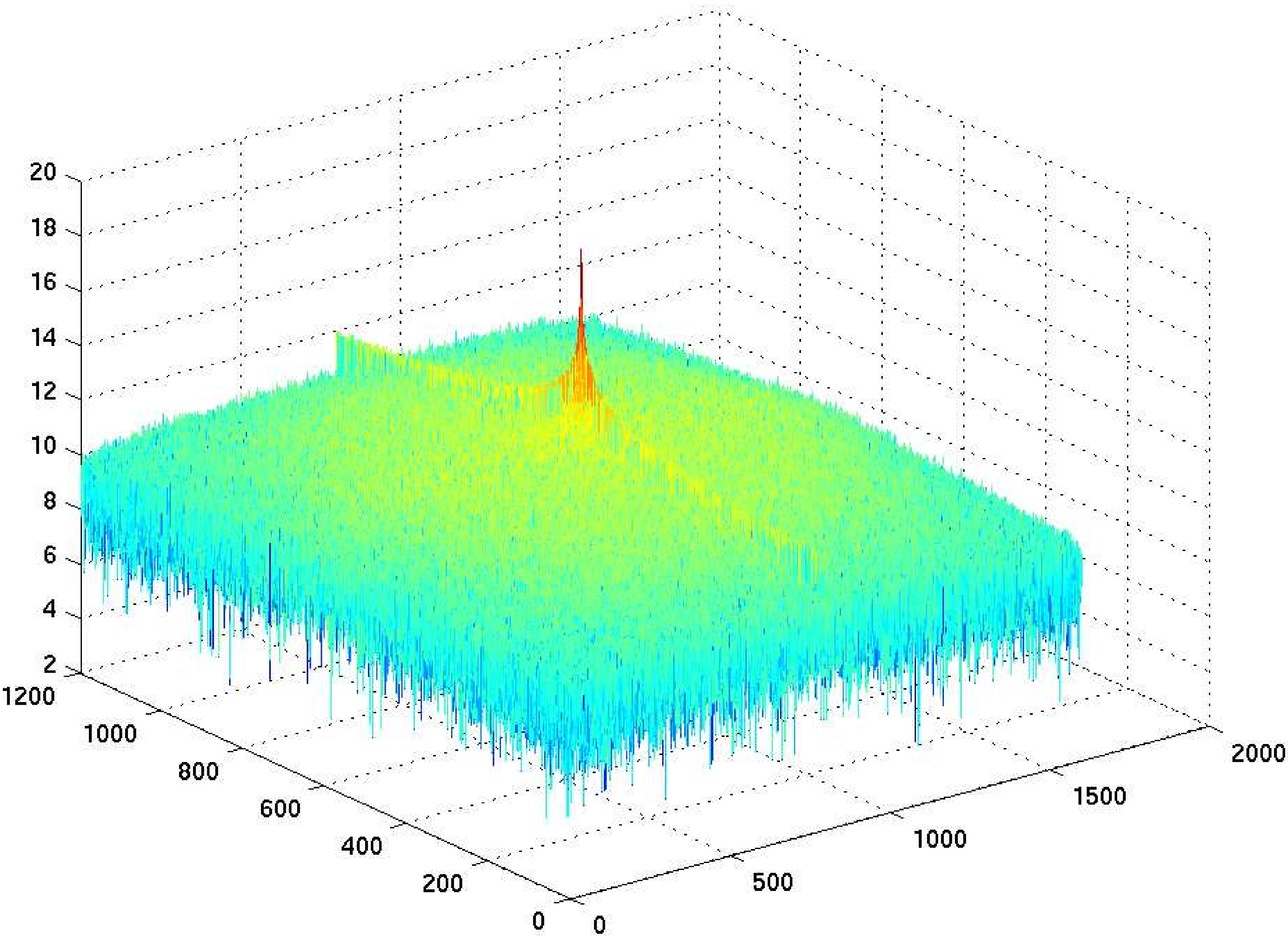



Low frequency image: log(abs(FFT2(im)))
200
400
600
800
1000
1200
200
400
600
800
1000
1200
1400
1600
18
16
14
12
10
8
6
4
2
0


Hi frequency image: log(abs(FFT2(im)))
200
400
600
800
1000
1200
200
400
600
800
1000
1200
1400
1600
18
16
14
12
10
8
6
4


Low frequency image: mesh print of log(abs(FFT2(im)))
20
15
10
5
0
-5
1200
1000
800
600
400
200
0
0
500
1000
1500
2000

Hi frequency image: mesh print of log(abs(FFT2(im)))



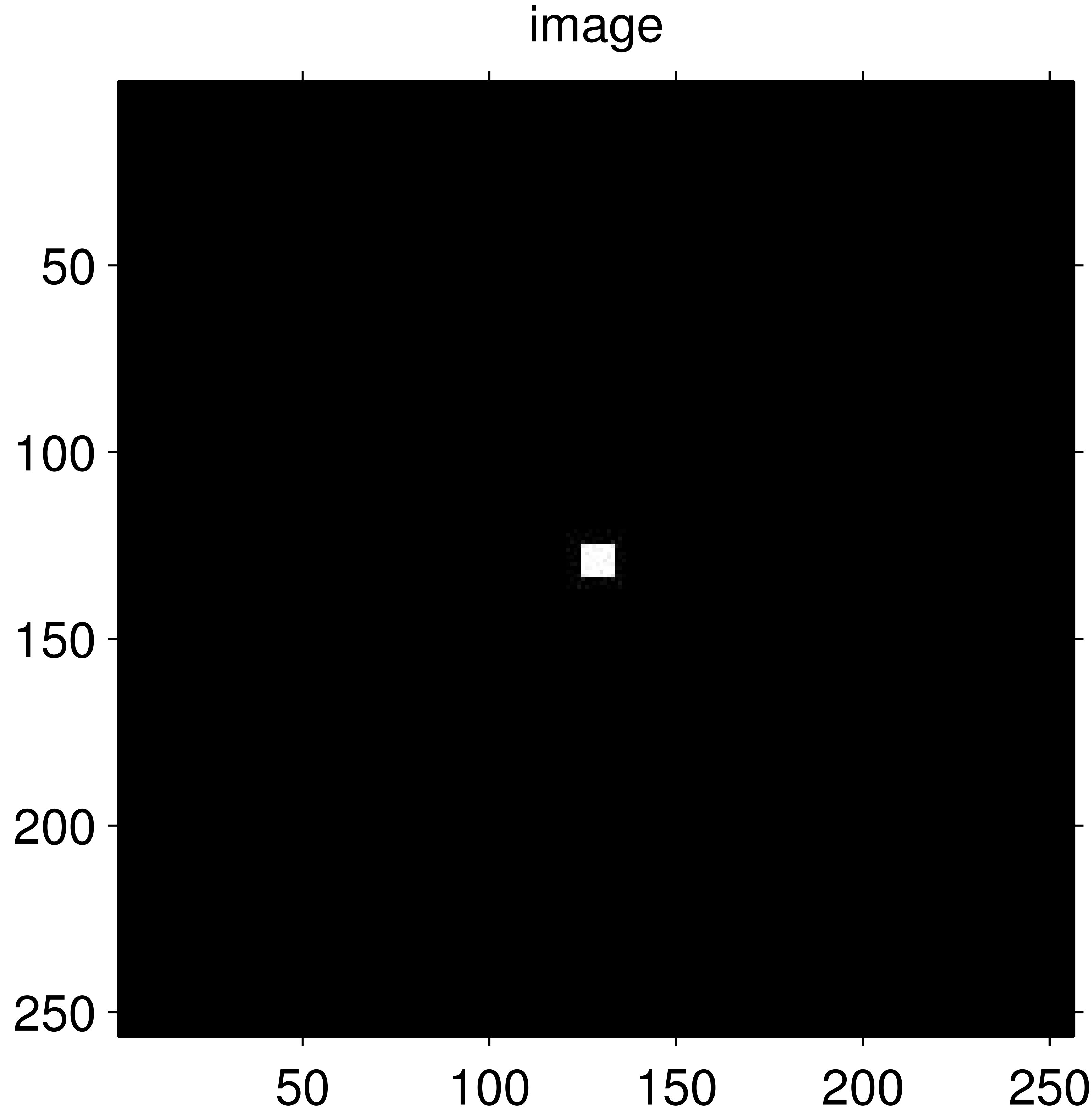
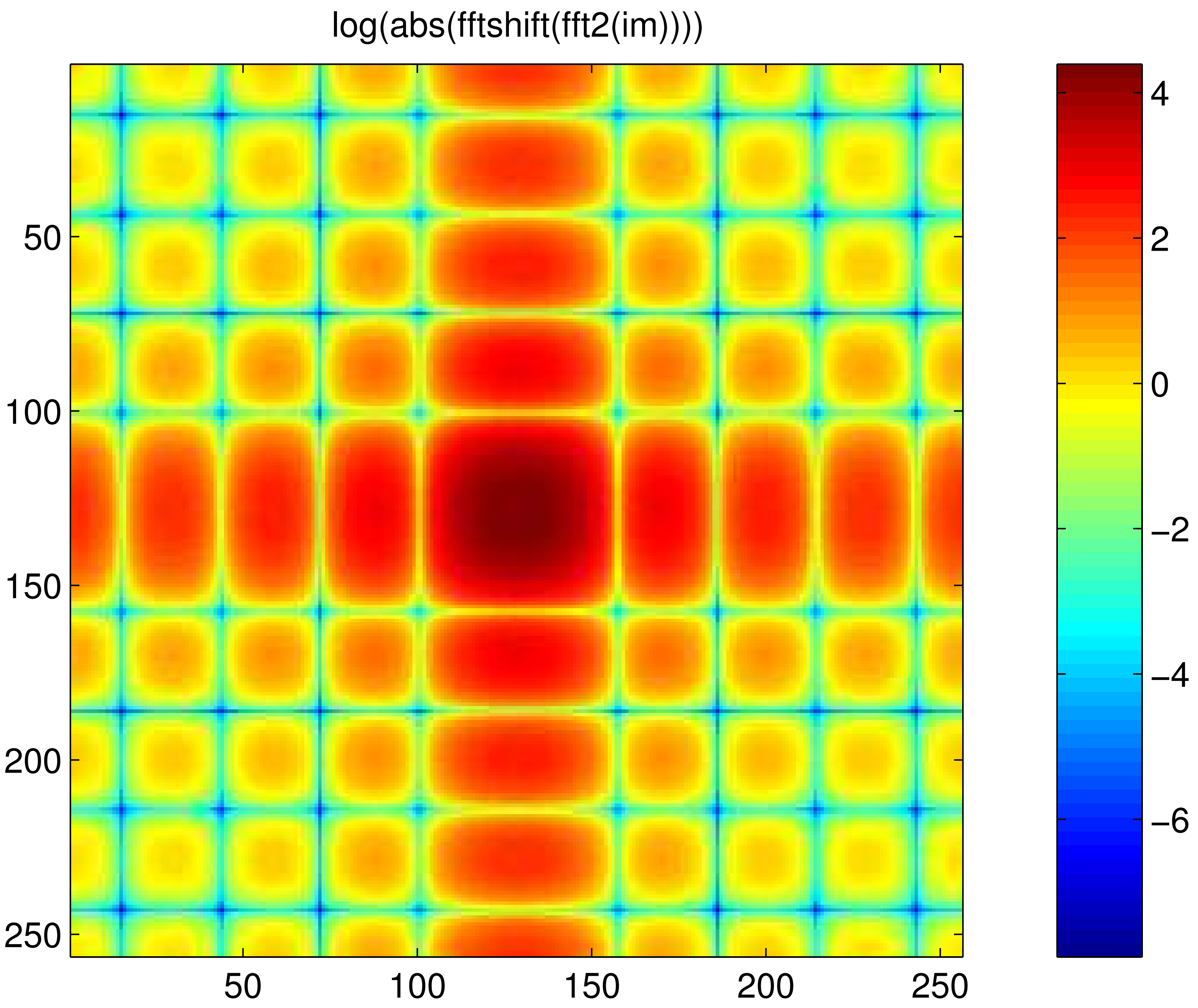


image
50
100
150
200
250
50
100
150
200
250



log(abs(fftshift(fft2(im))))
50
100
150
200
250
50
100
150
200
250
4
2
0
−2
−4
−6

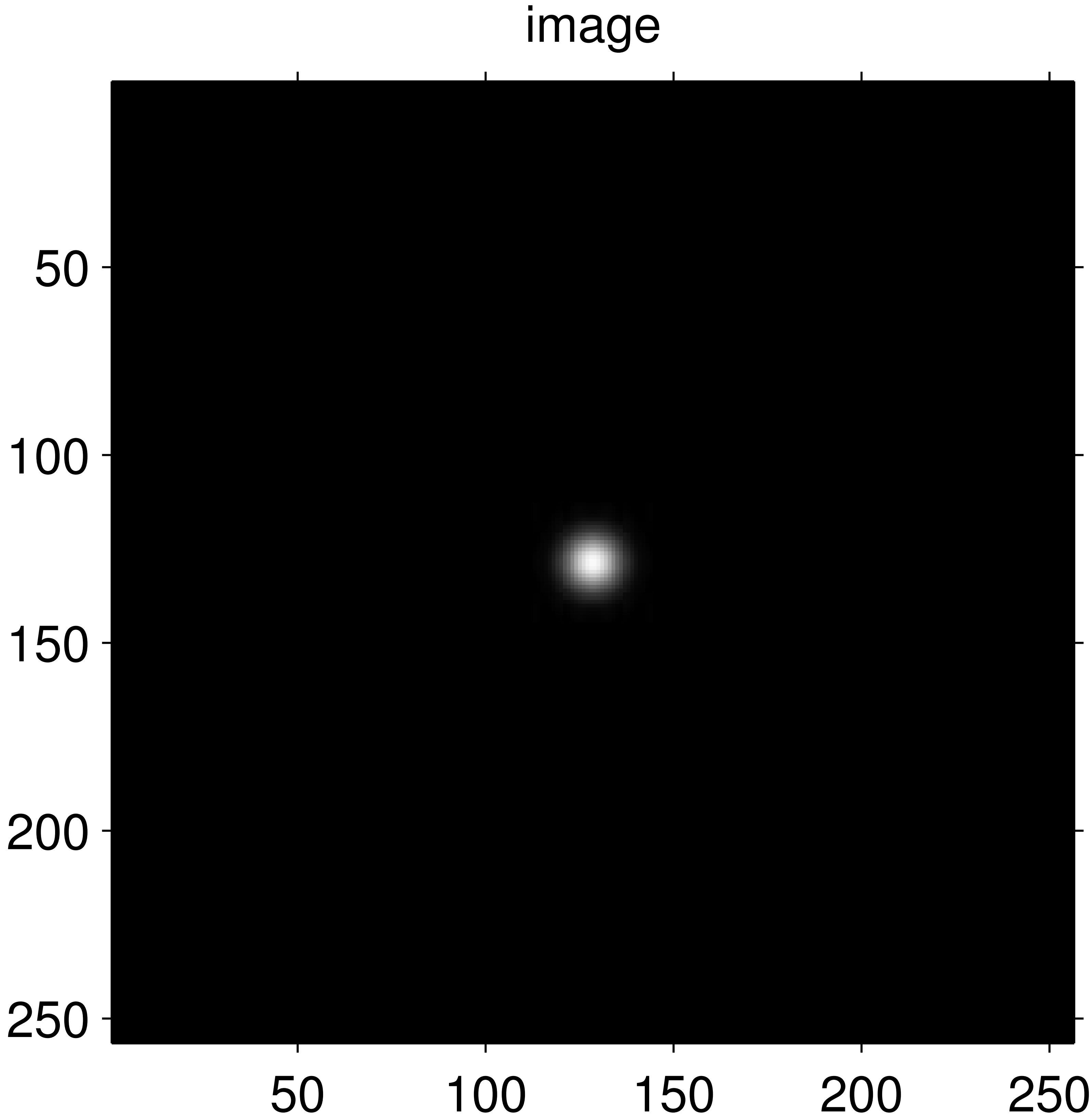
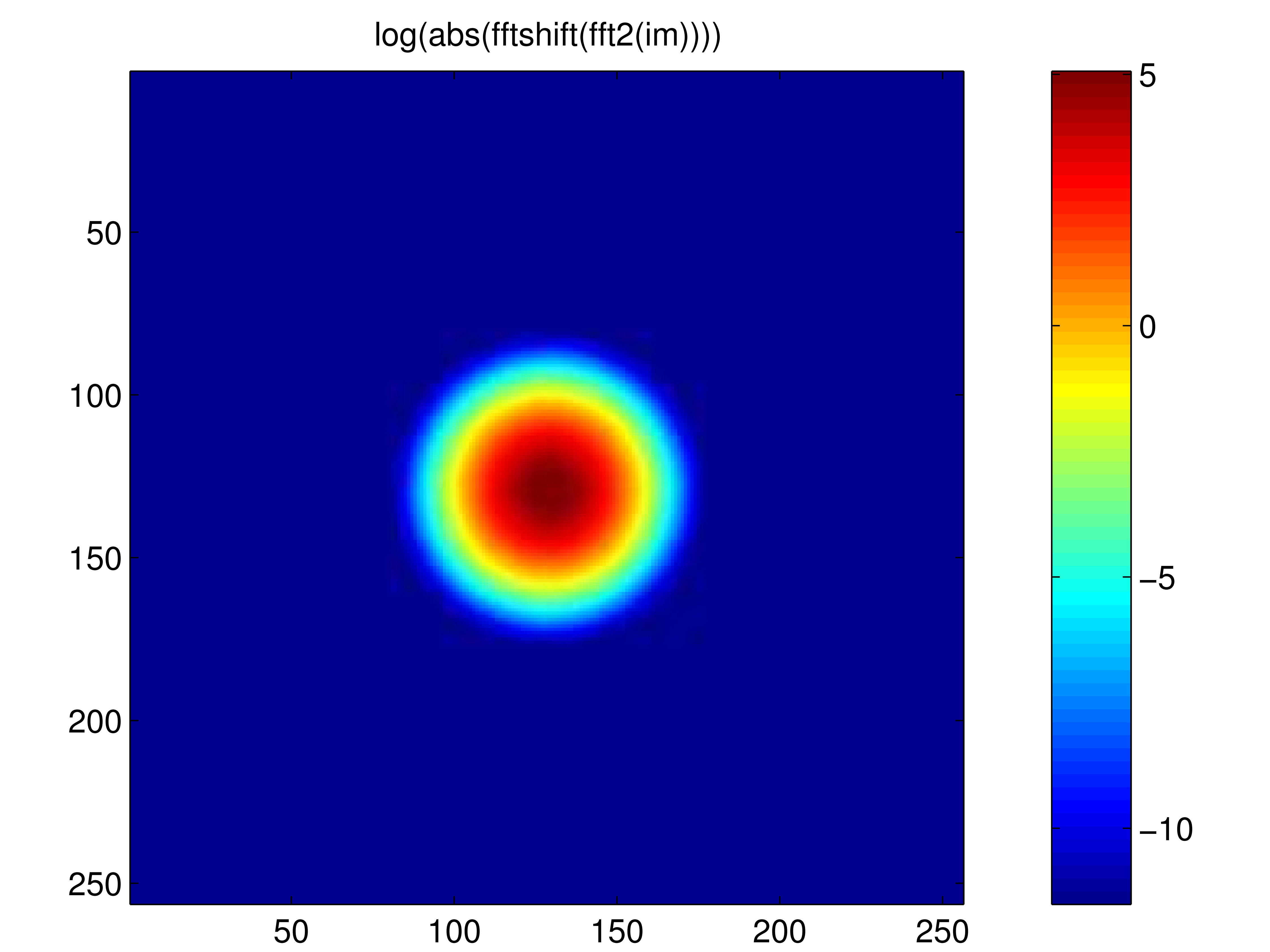


image
50
100
150
200
250
50
100
150
200
250


log(abs(fftshift(fft2(im))))
50
100
150
200
250
50
100
150
200
250
5
0
−5
−10

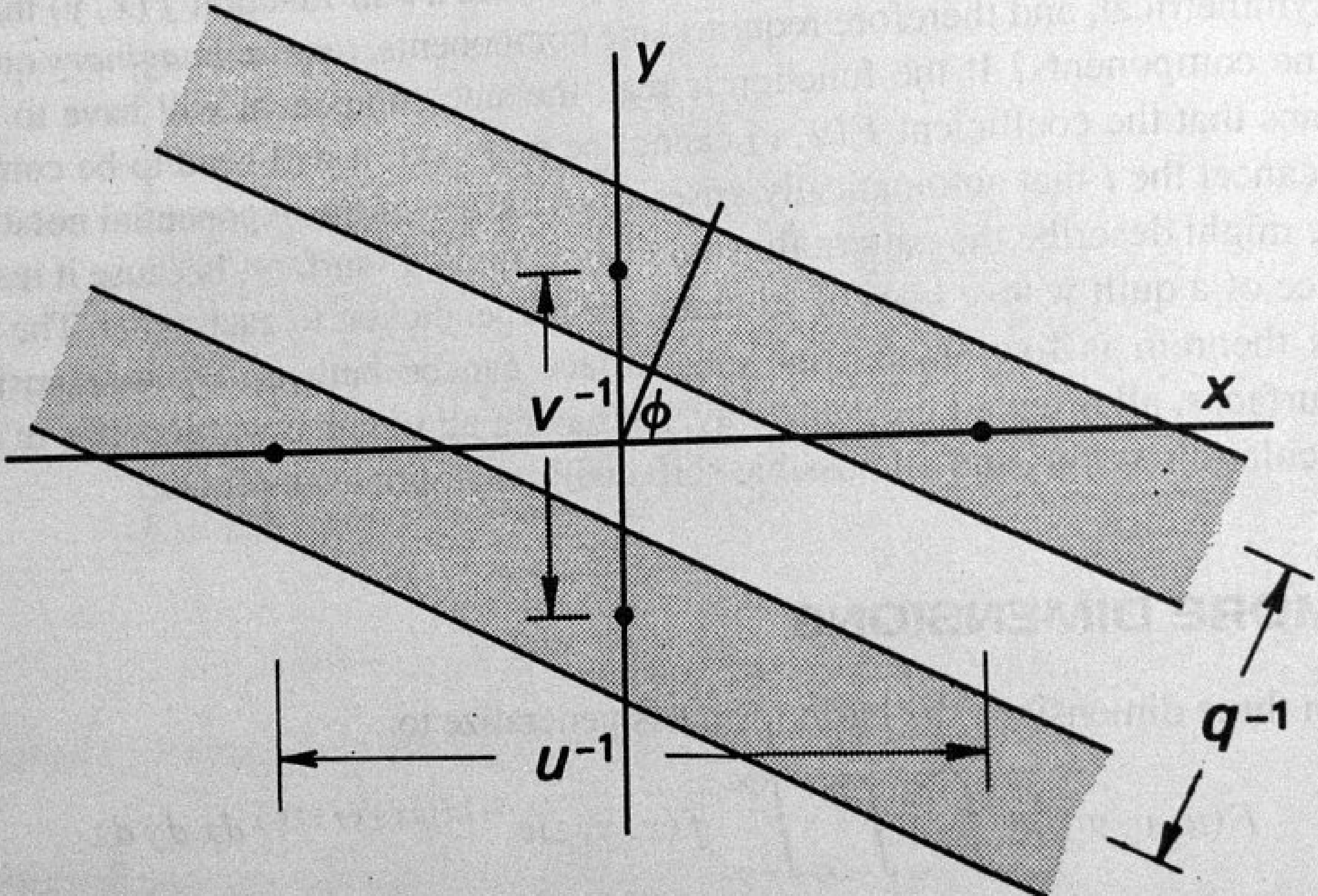


Figure 4-2 A "corrugation" $\cos[2\pi(ux + vy)]$. The shaded zones are negative. The spatial frequency in the x-direction is u (in the y-direction, v); *the* spatial frequency is q.

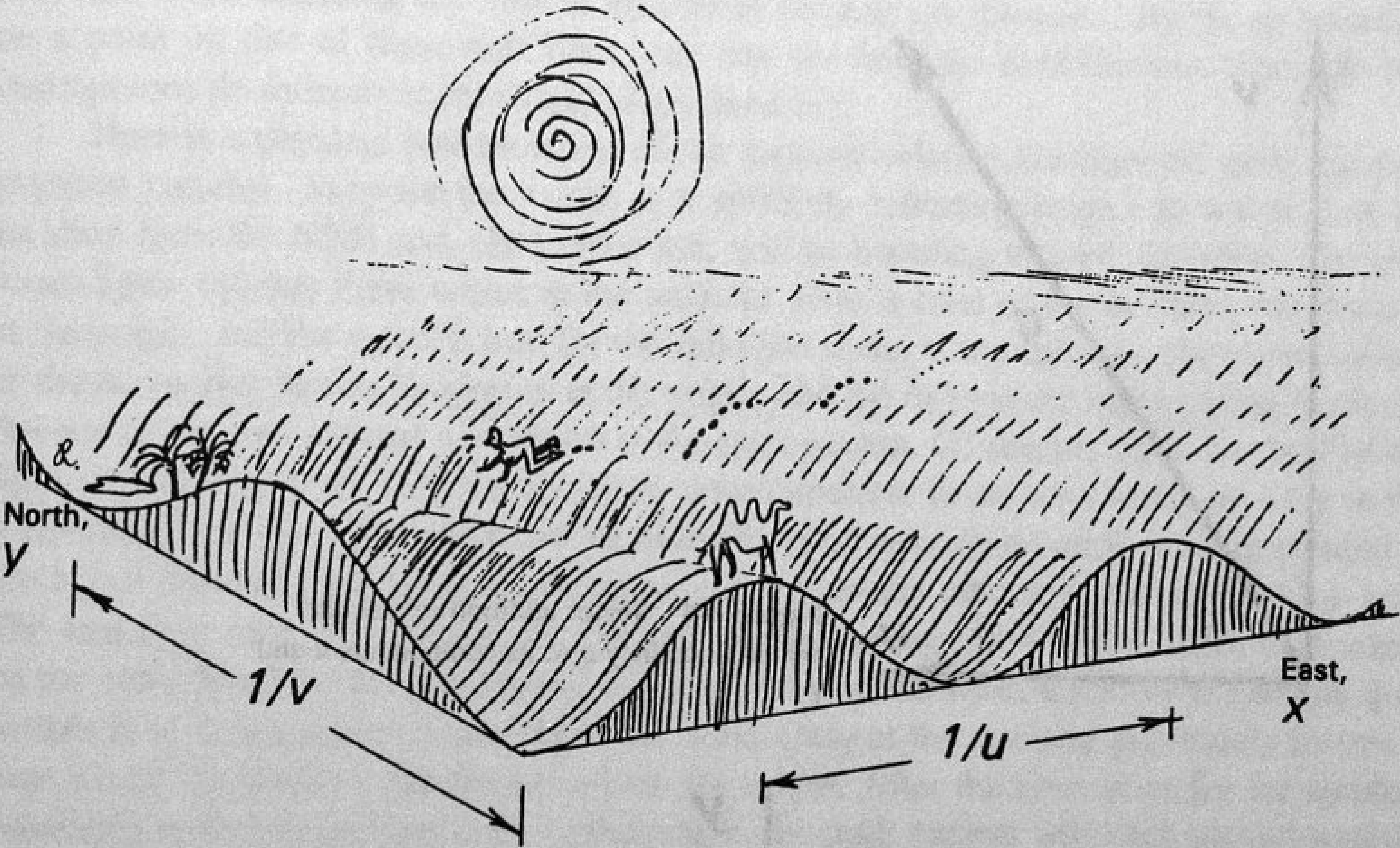
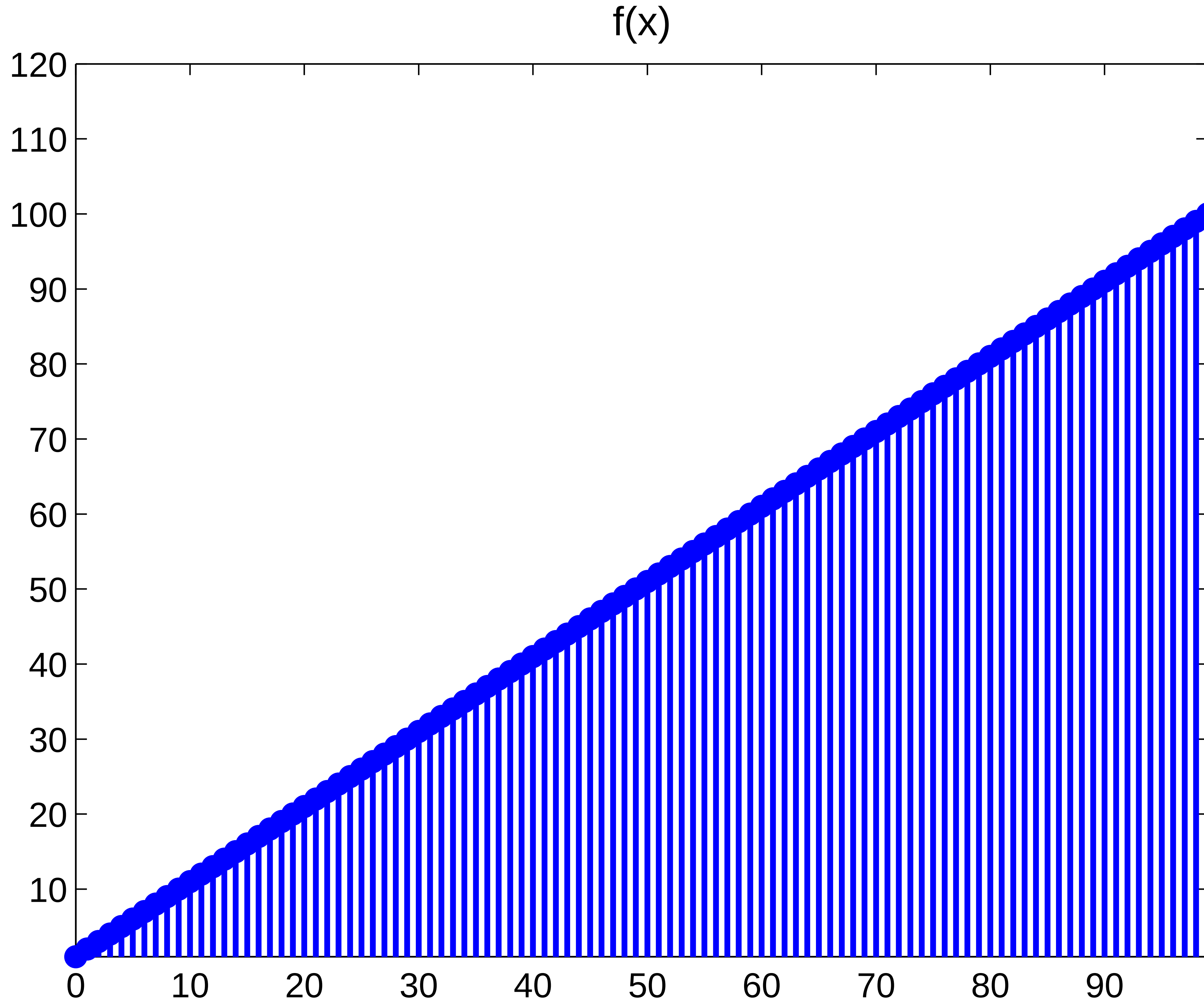
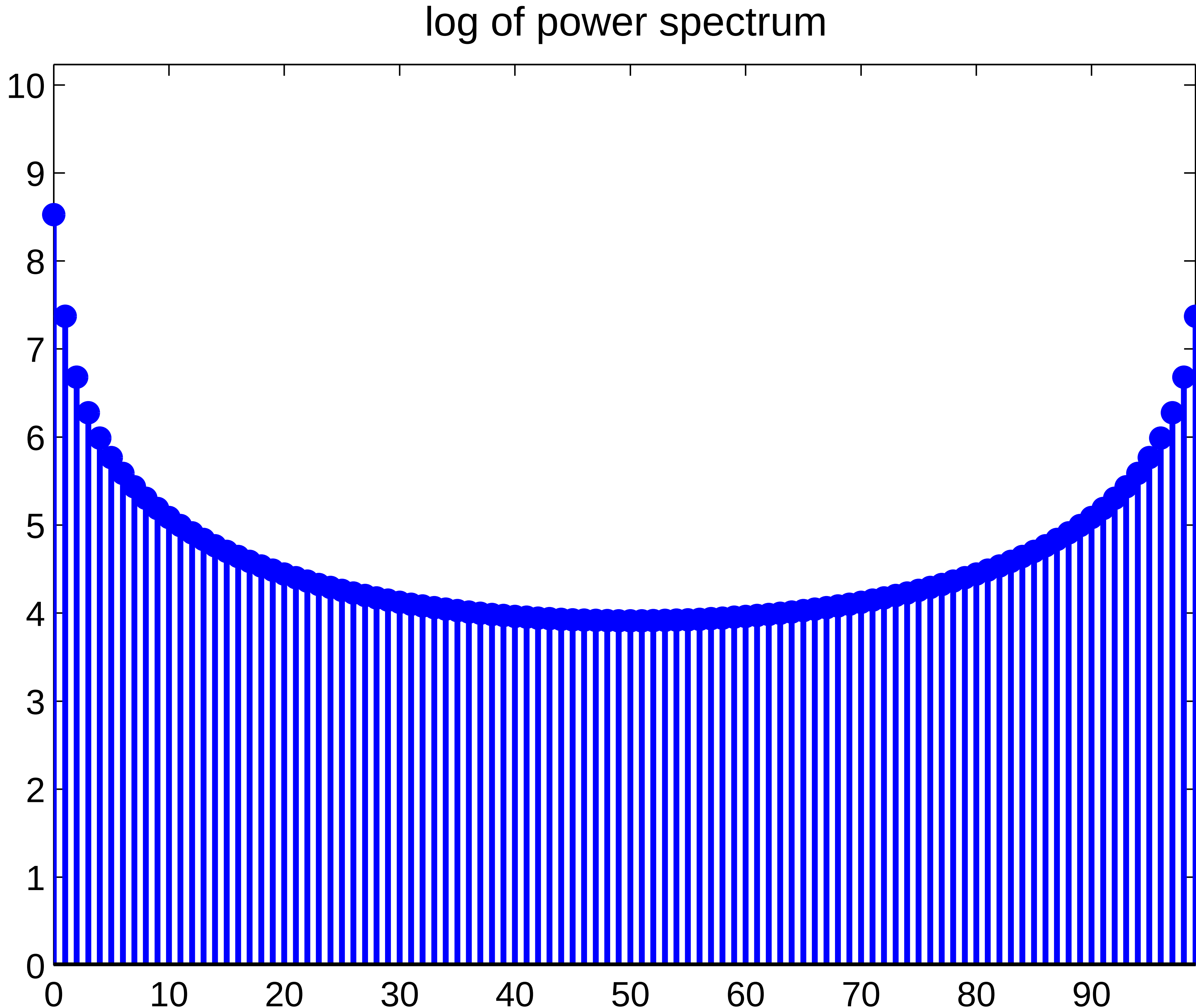
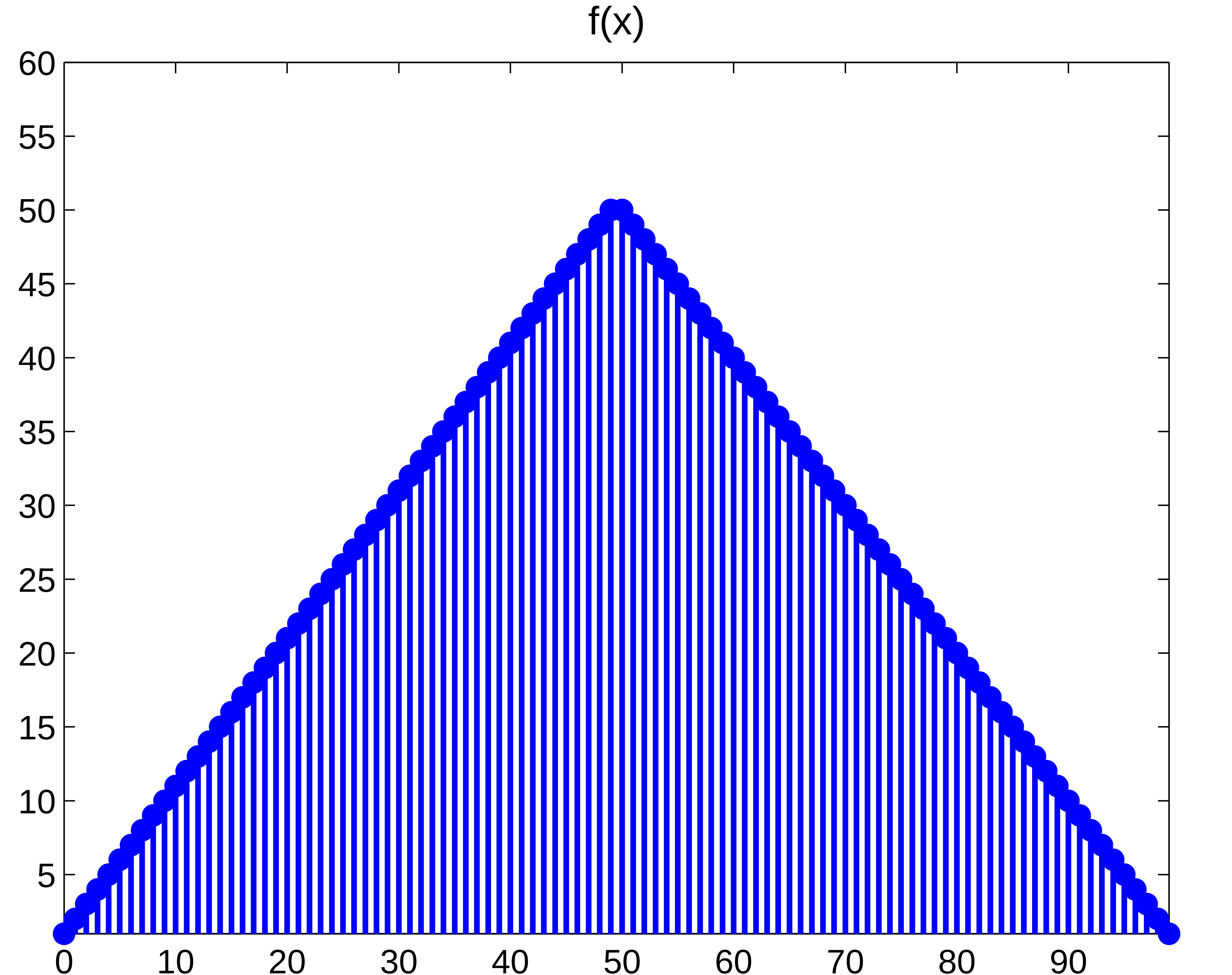


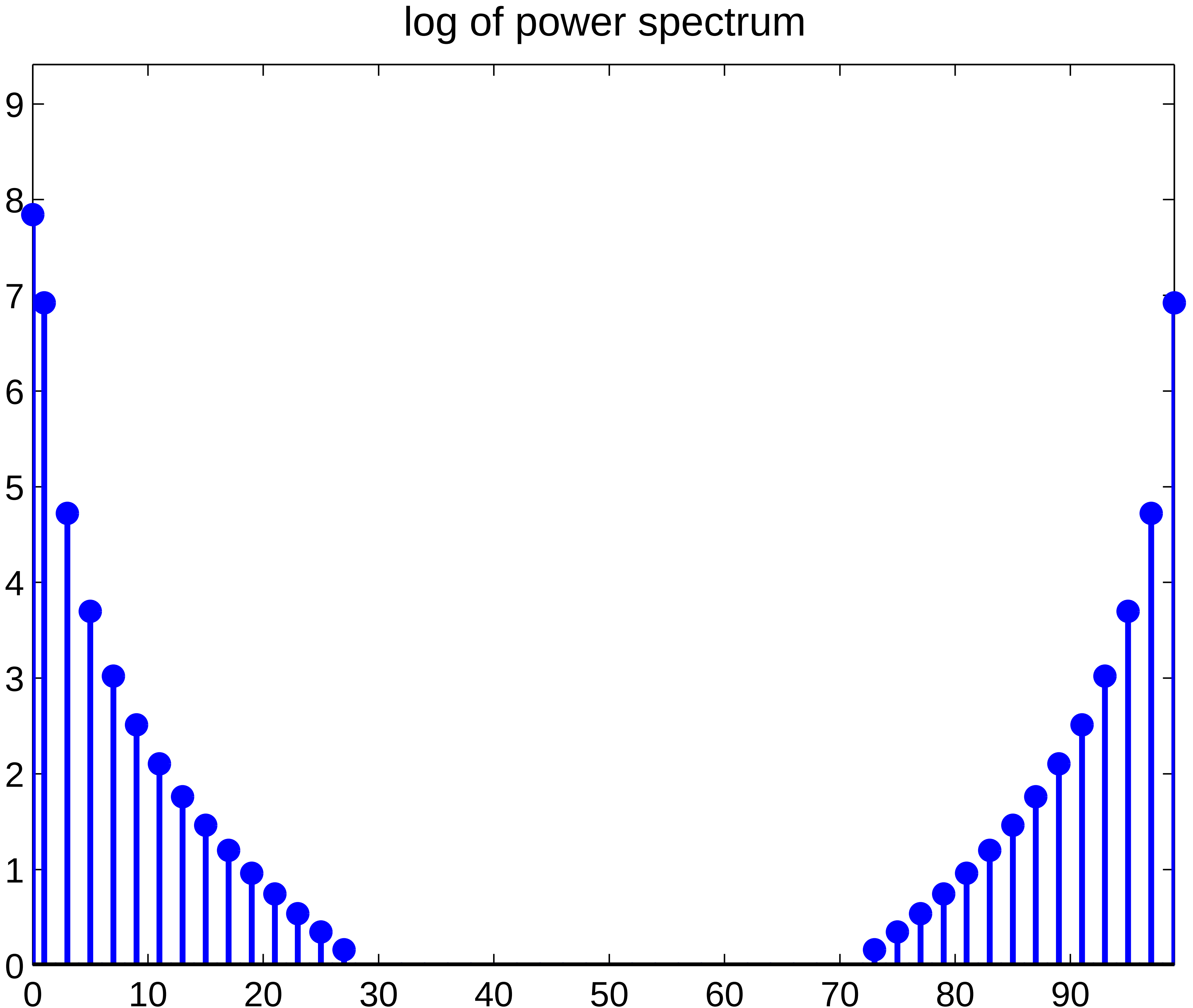
Figure 4-3 The x-component of spatial frequency is u, measured in cycles per unit of x. The spatial period, or crest-to-crest distance traveling east, is u^{-1} (see camel). The spatial frequency q is measured in the direction of hardest going (see man).


f(x)
120
110
100
90
80
70
60
50
40
30
20
10
0
10
20
30
40
50
60
70
80
90

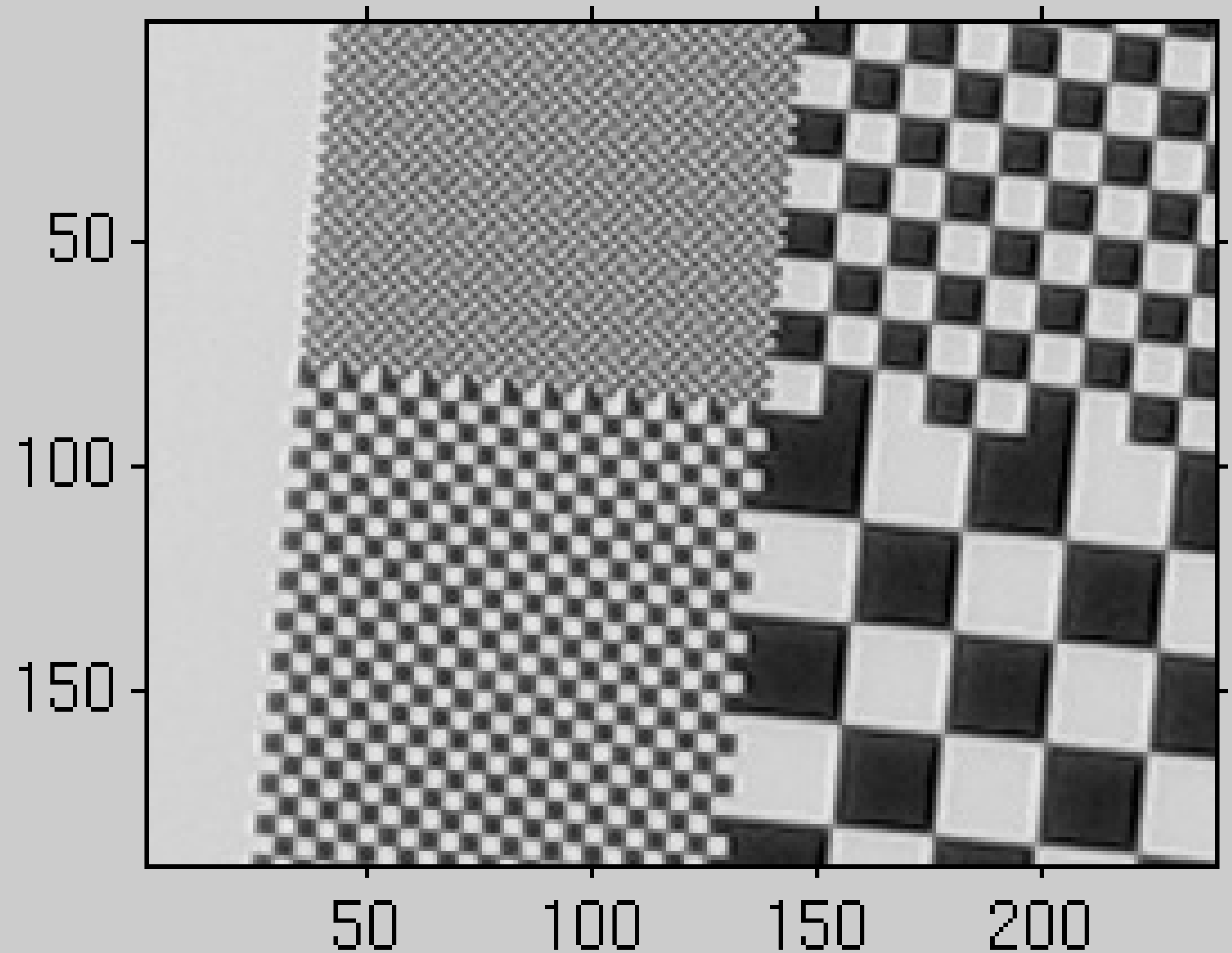

log of power spectrum
10
9
8
7
6
5
4
3
2
1
0
0
10
20
30
40
50
60
70
80
90



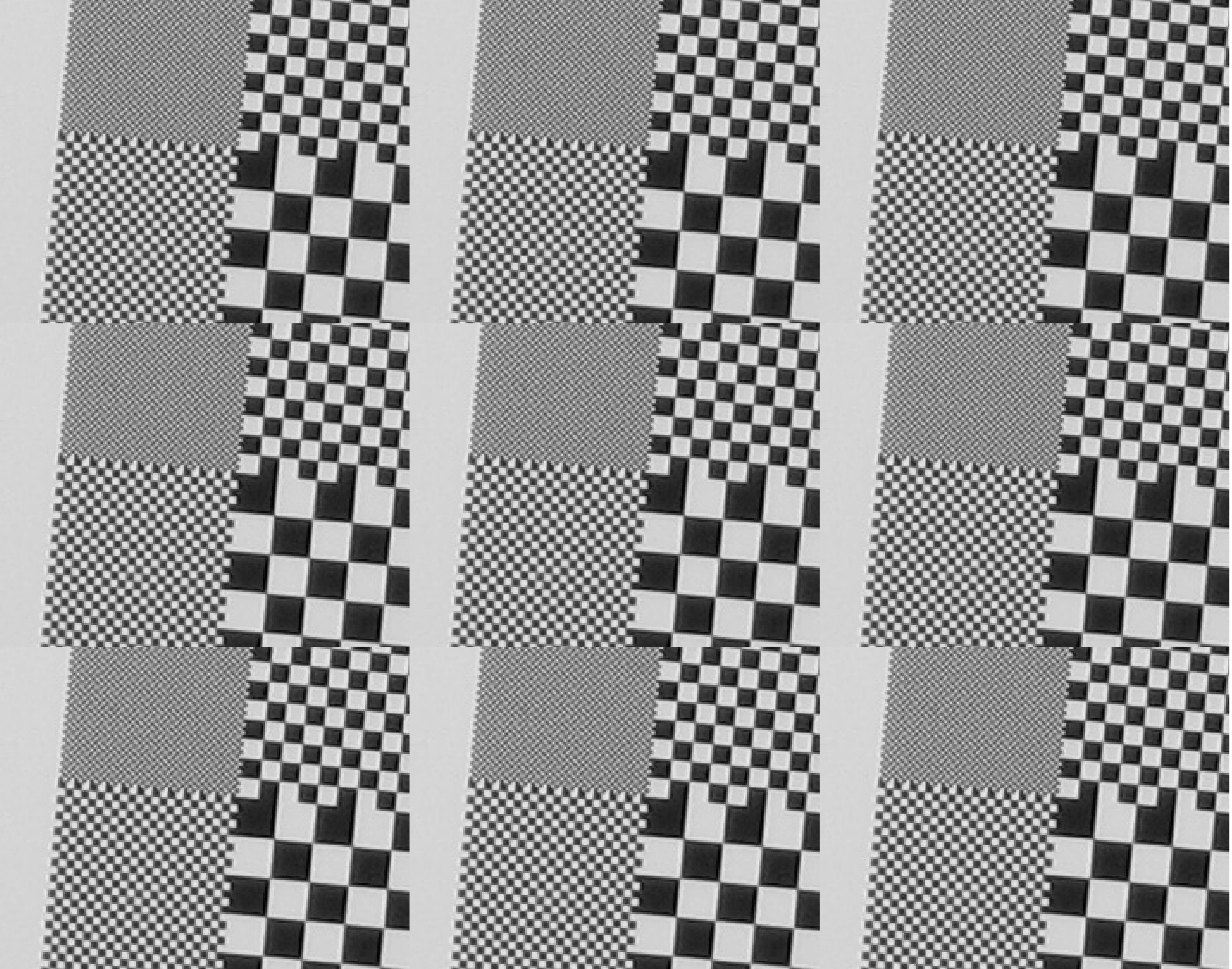
f(x)
60
55
50
45
40
35
30
25
20
15
10
5
0
10
20
30
40
50
60
70
80
90

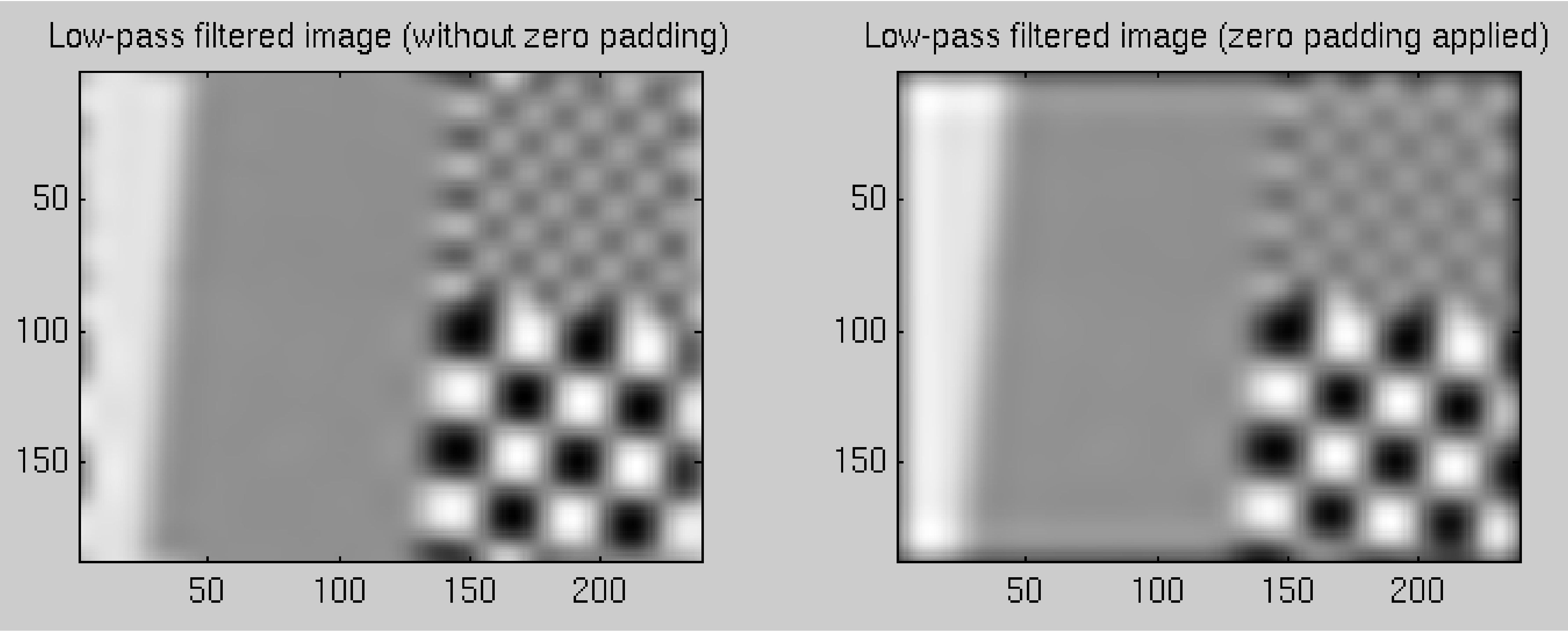


log of power spectrum
0
1
2
3
4
5
6
7
8
9
0
10
20
30
40
50
60
70
80
90

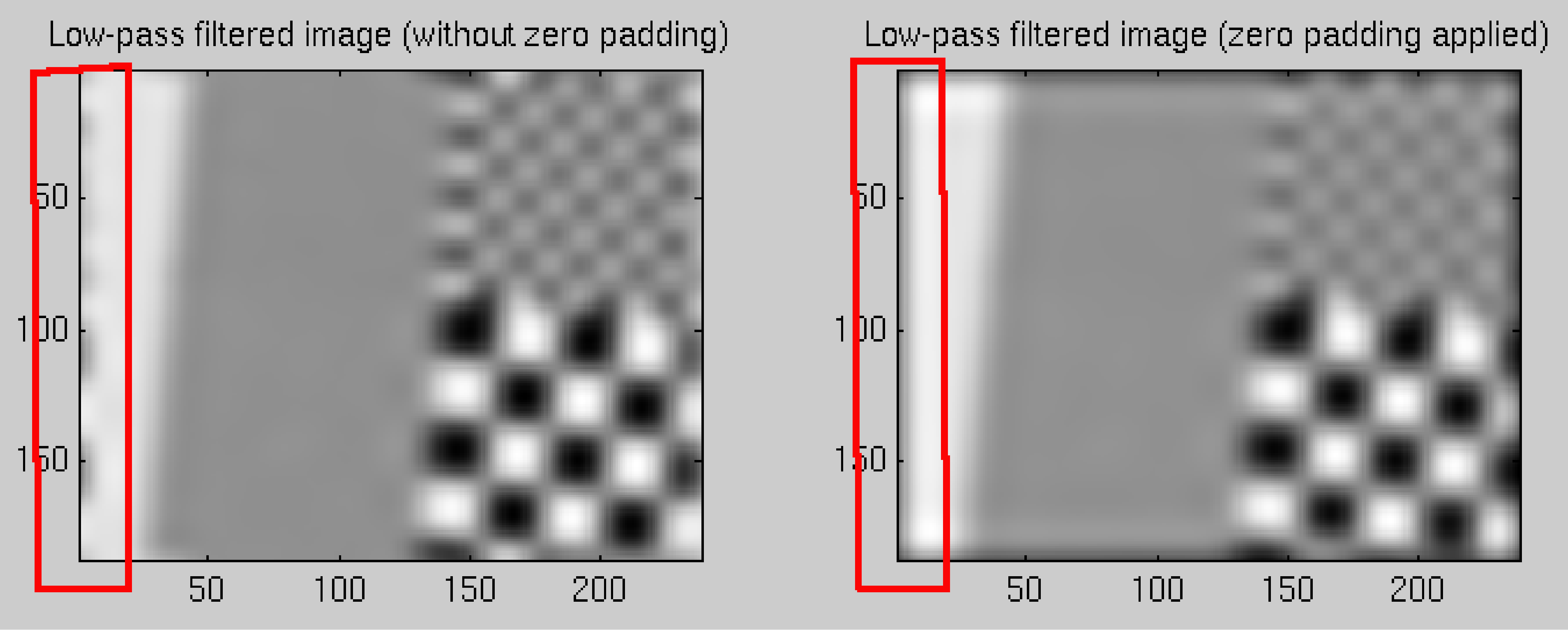


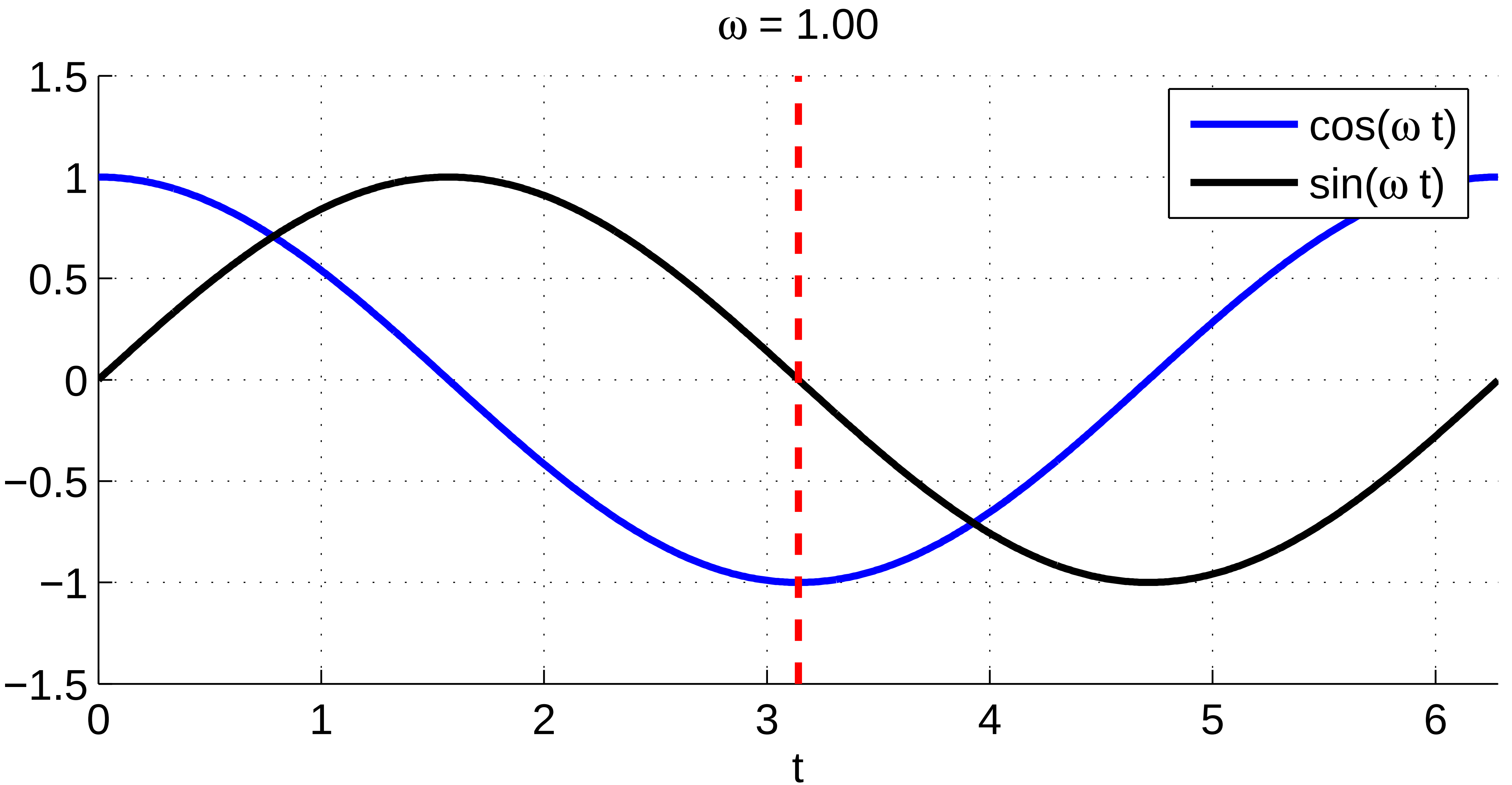
50
100
150
50
100
150
200

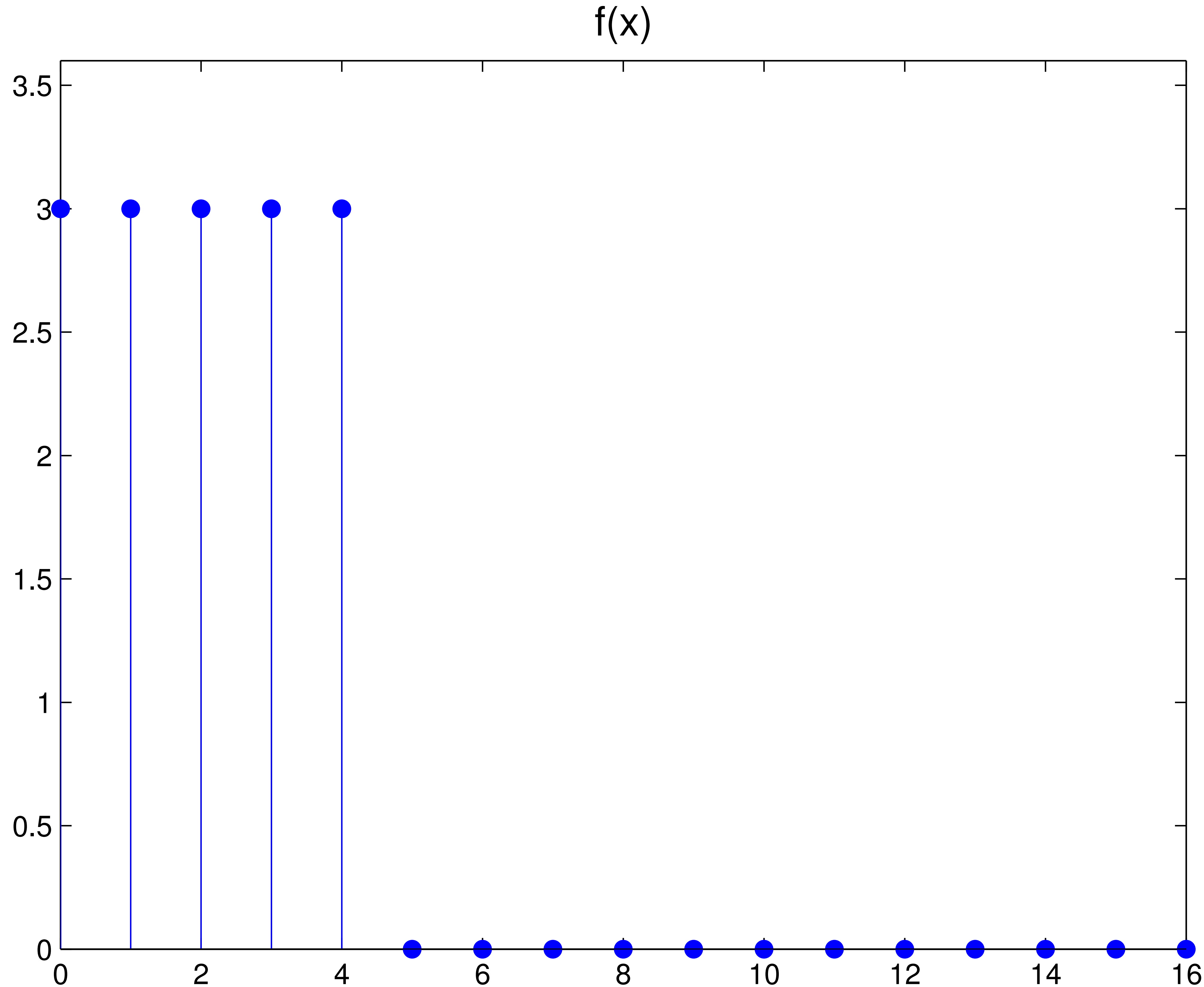


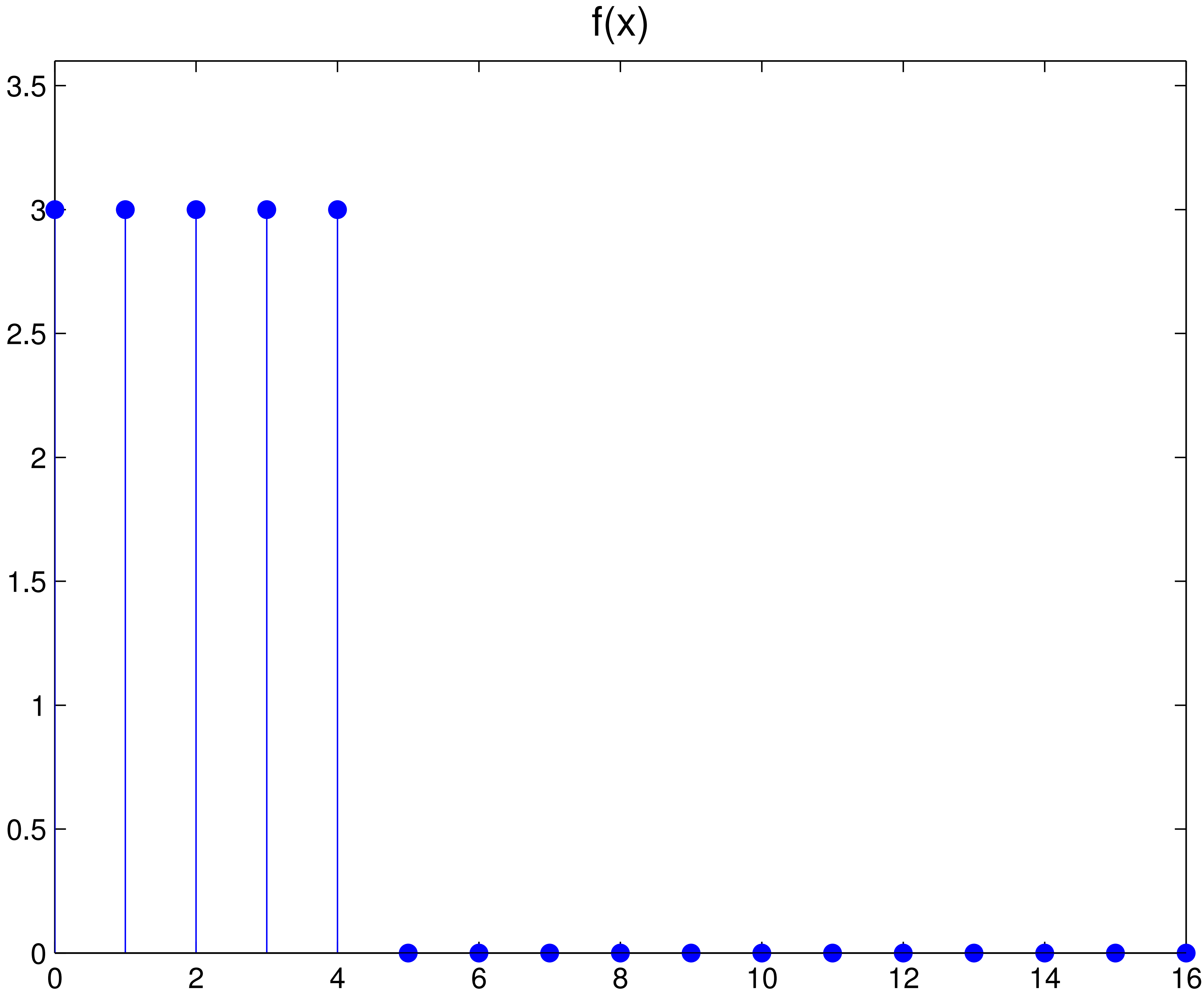


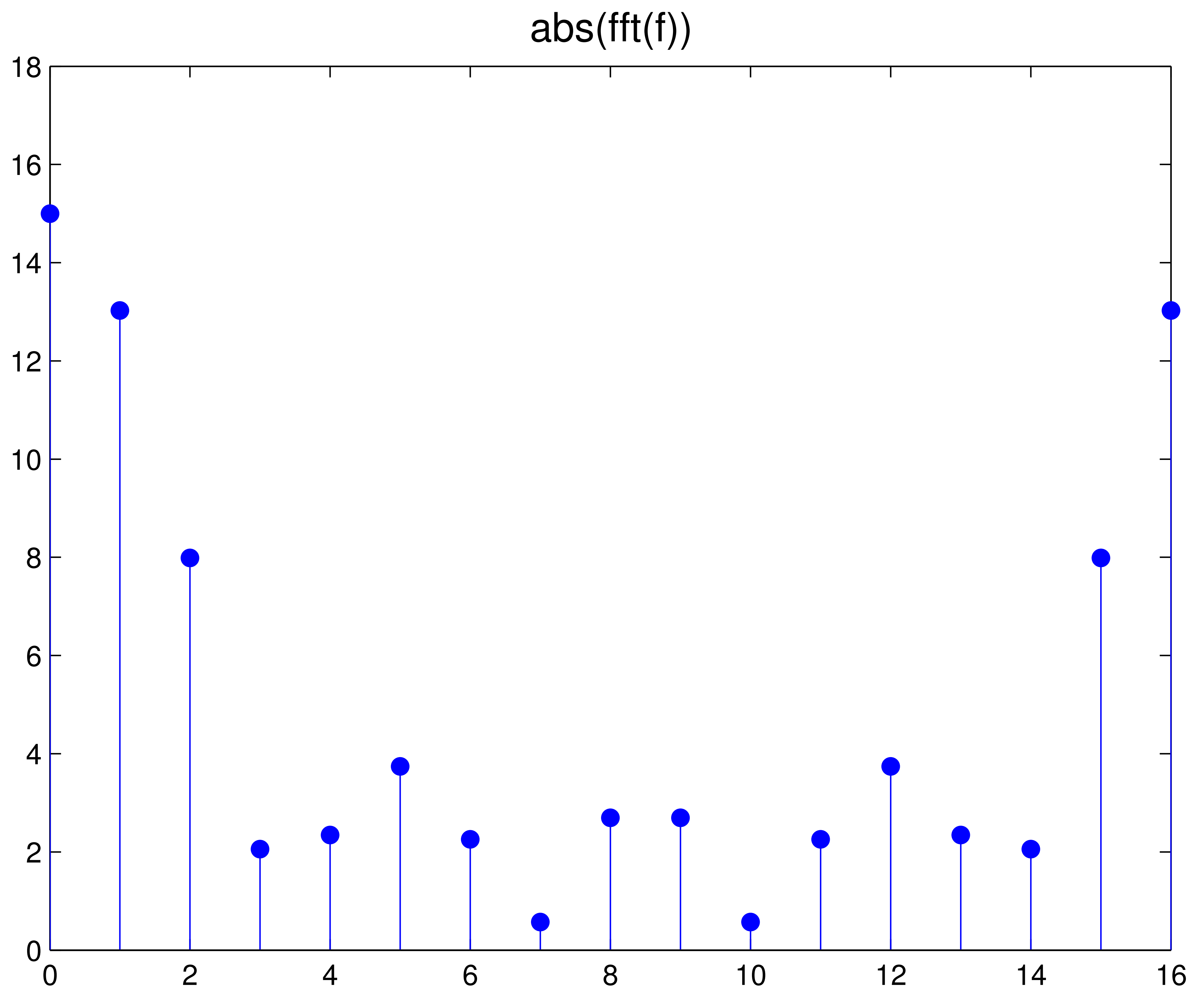
Low-pass filtered image (without zero padding)
Low-pass filtered image (zero padding applied)
50
100
150
200

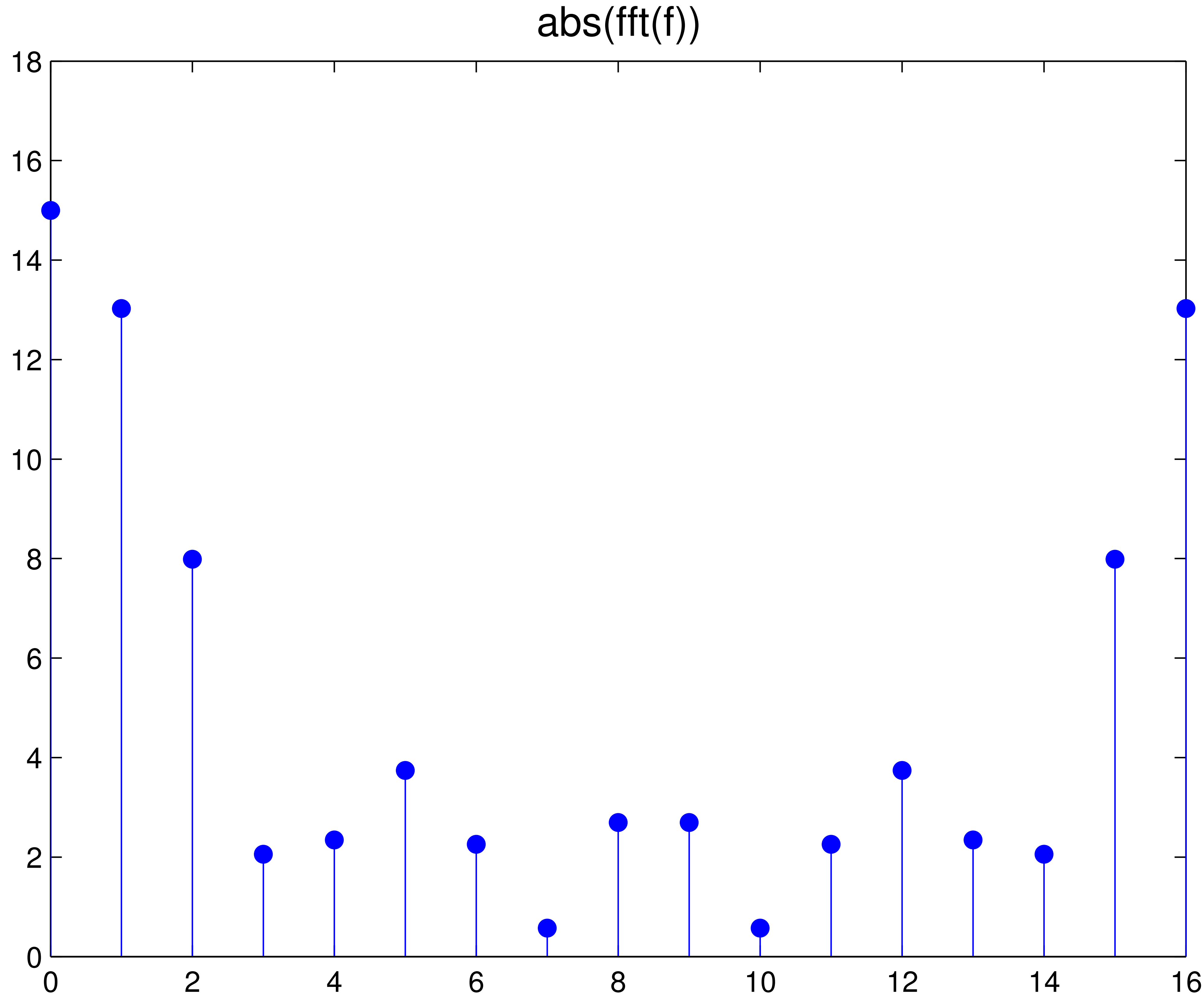

Low-pass filtered image (without zero padding)
50
100
150
50
100
150
200
Low-pass filtered image (zero padding applied)
50
100
150
50
100
150
200

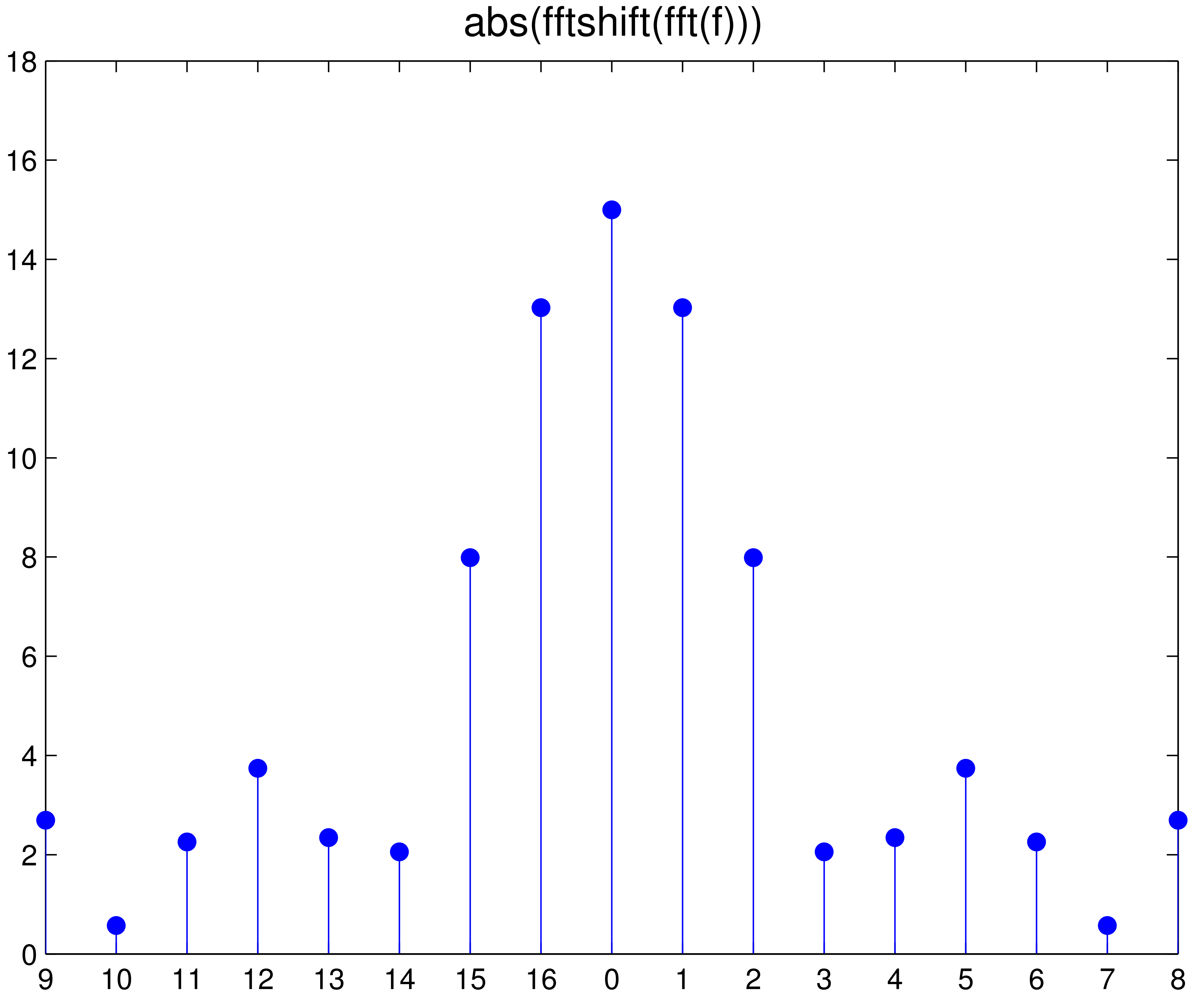

ω = 1.00
cos(ω t)
sin(ω t)
1.5
1
0.5
0
−0.5
−1
−1.5
0
1
2
3
4
5
6
t


f(x)
3.5
3
2.5
2
1.5
1
0.5
0
0
2
4
6
8
10
12
14
16


f(x)
3.5
3
2.5
2
1.5
1
0.5
0
0
2
4
6
8
10
12
14
16


abs(fft(f))
0
2
4
6
8
10
12
14
16
18
0
2
4
6
8
10
12
14
16


abs(fft(f))
0
2
4
6
8
10
12
14
16
18



abs(fftshift(fft(f)))
0
2
4
6
8
10
12
14
16
18
9
10
11
12
13
14
15
16
0
1
2
3
4
5
6
7
8

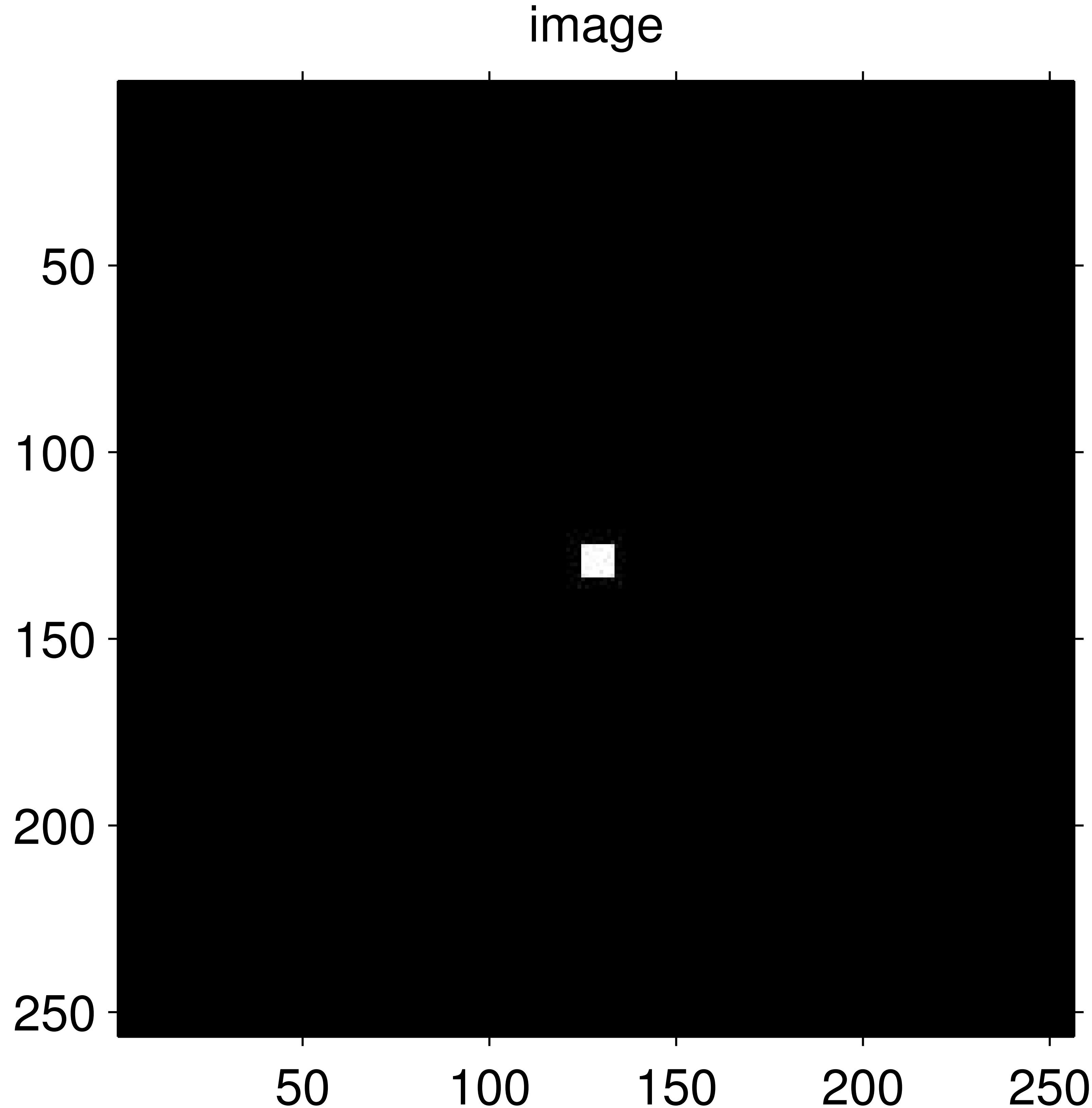
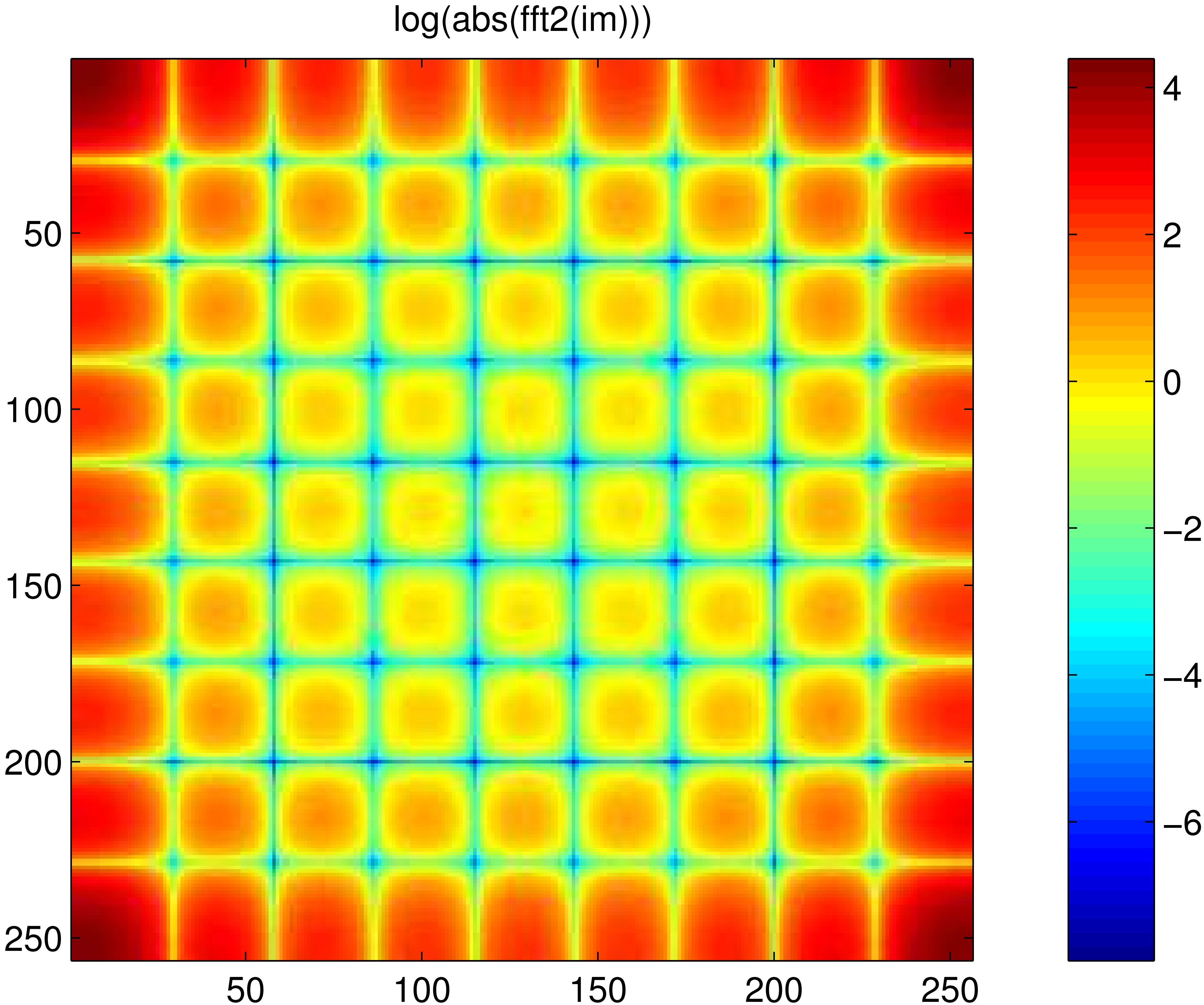


image
50
100
150
200
250
50
100
150
200
250



log(abs(fft2(im)))
50
100
150
200
250
50
100
150
200
250
4
2
0
−2
−4
−6

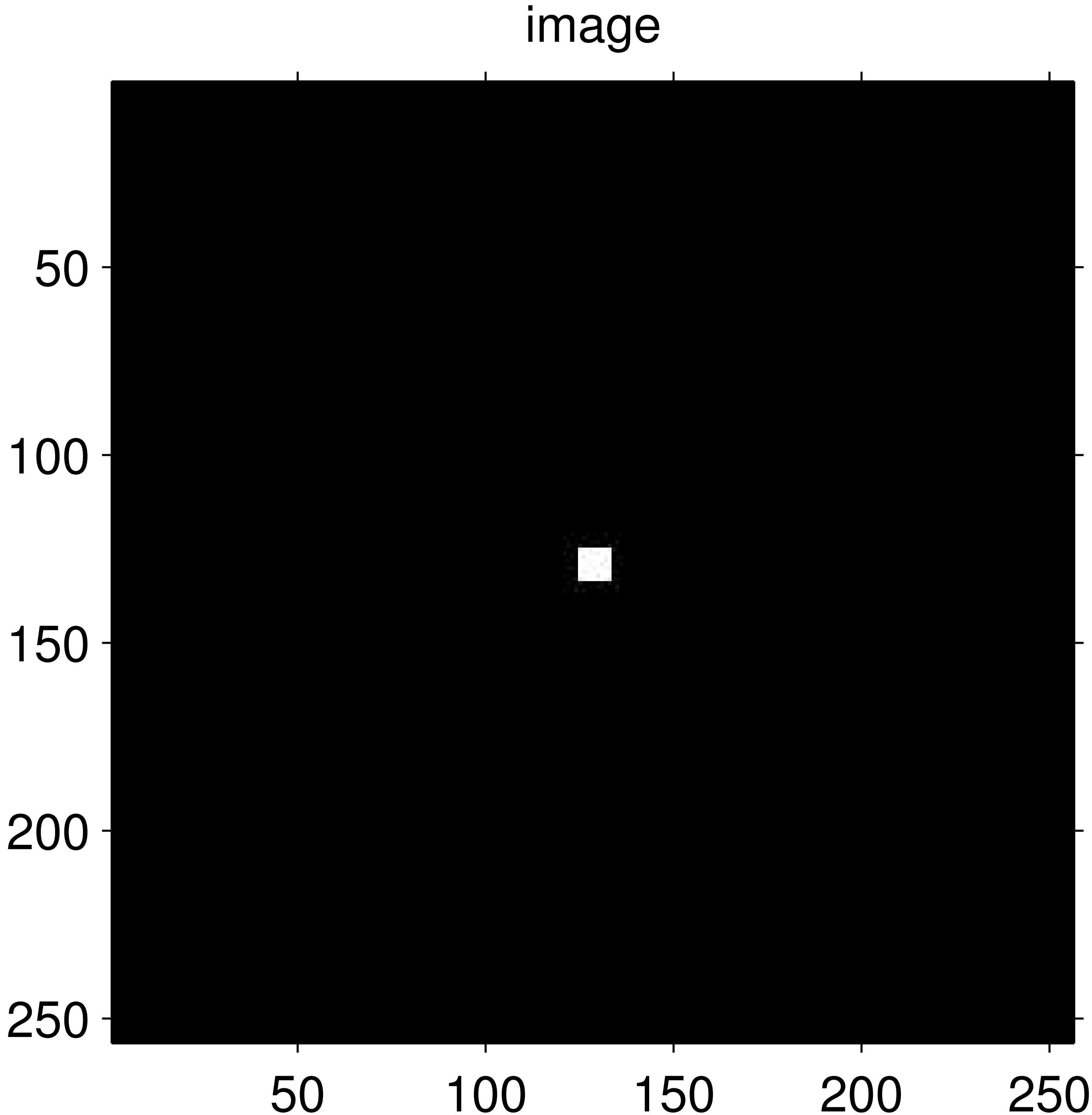
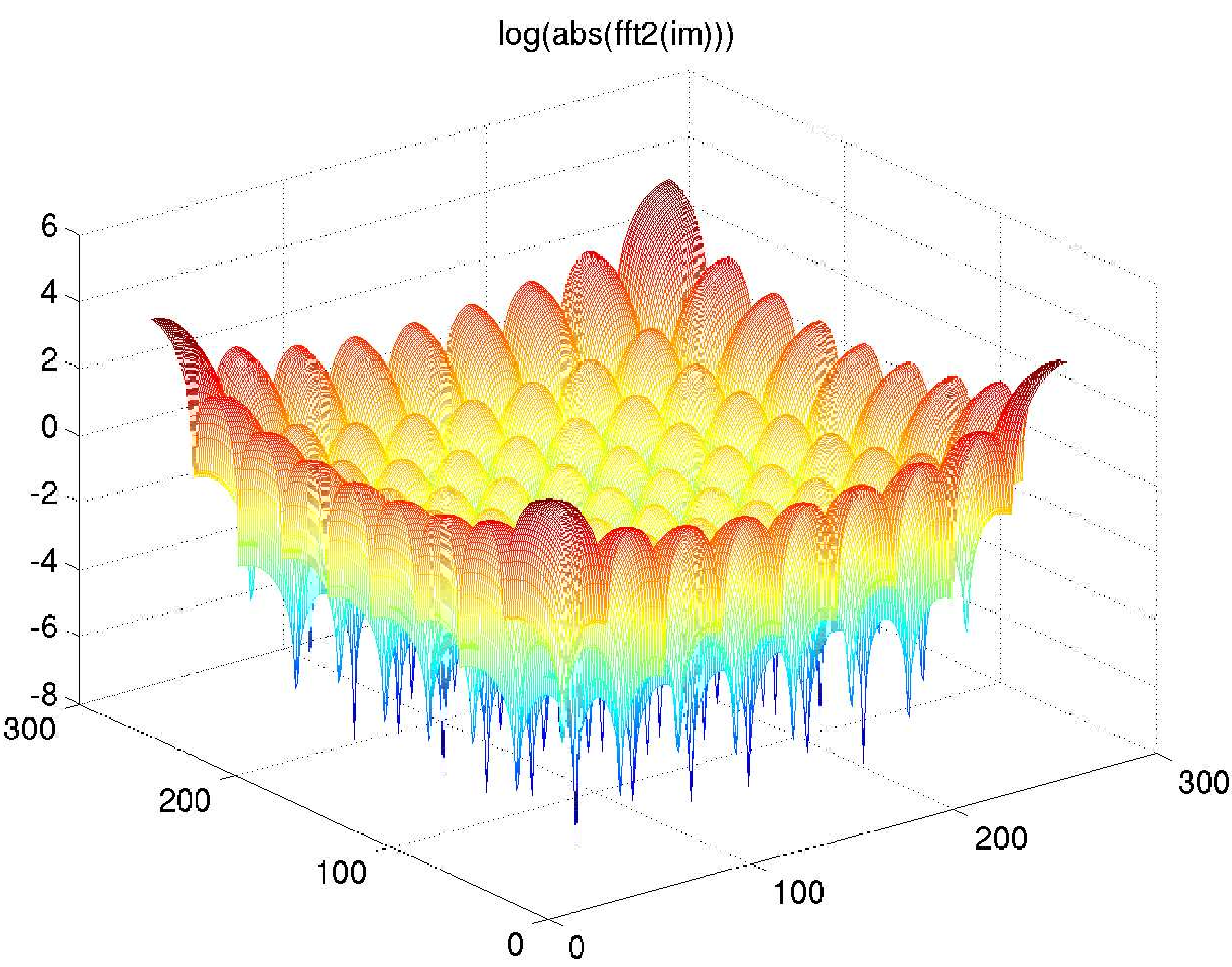


image
50
100
150
200
250
50
100
150
200
250


log(abs(fft2(im)))
6
4
2
0
-2
-4
-6
-8
300
200
100
0
0
100
200
300

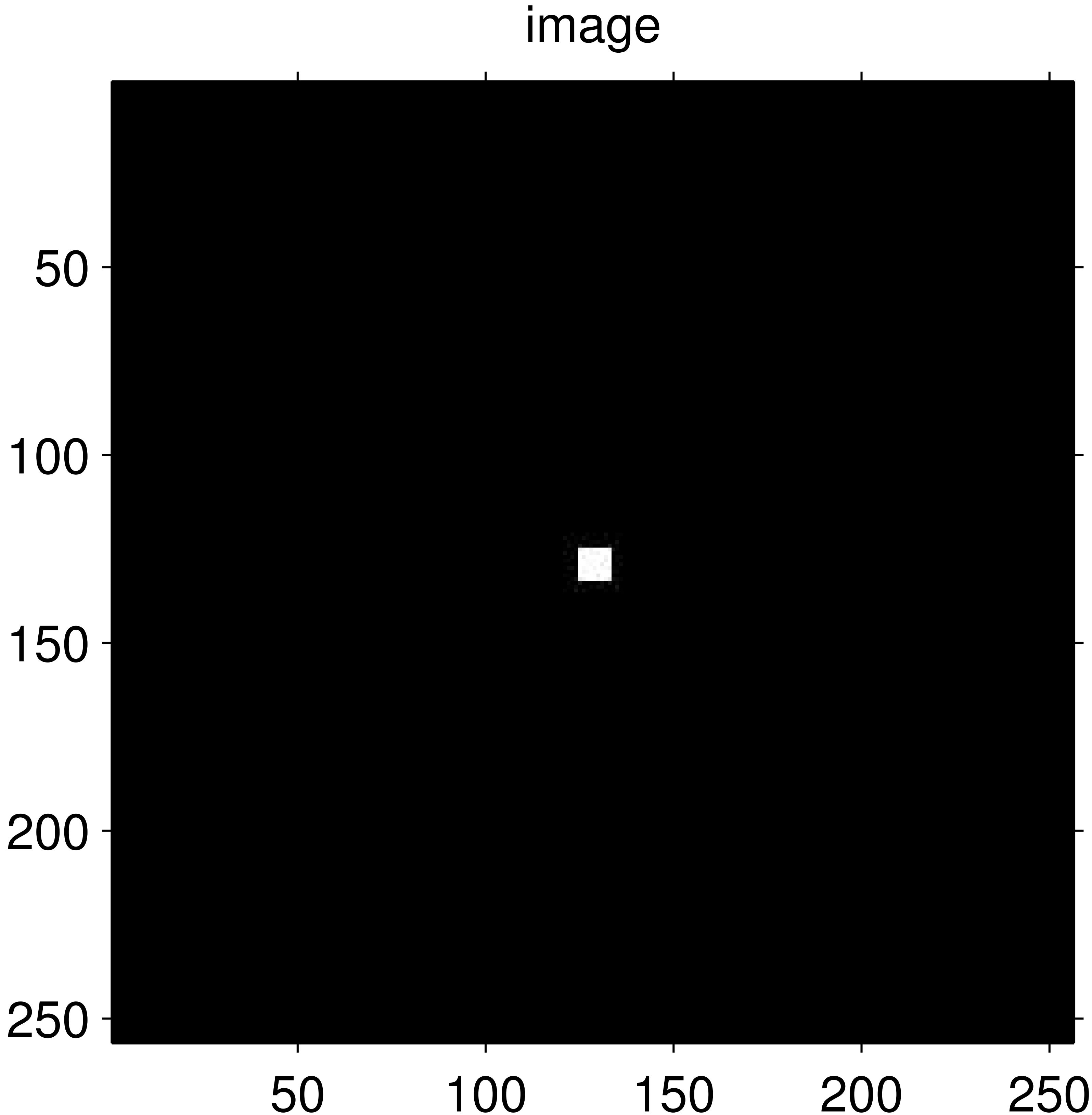
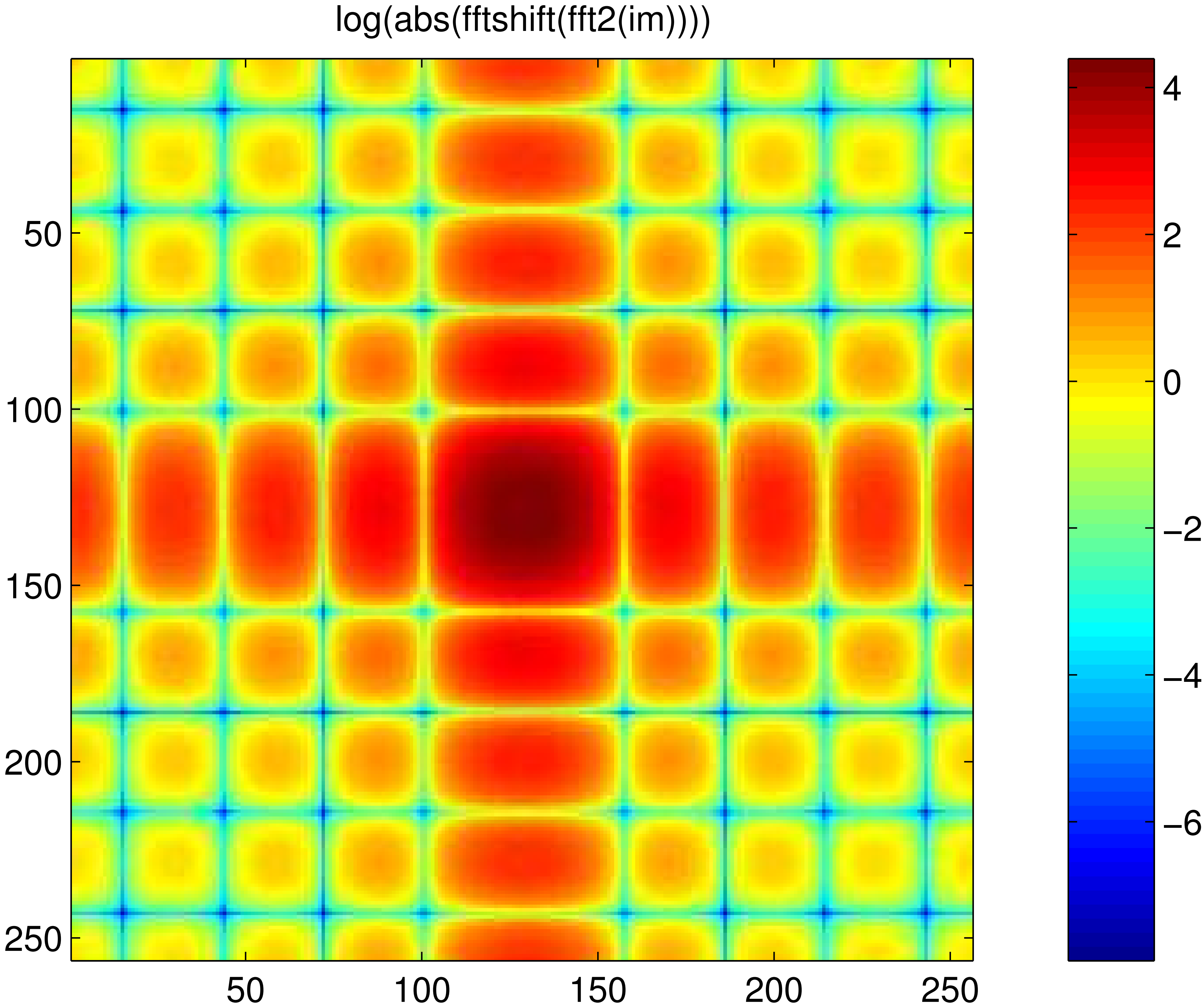


image
50
100
150
200
250
50
100
150
200
250


log(abs(fftshift(fft2(im))))
50
100
150
200
250
50
100
150
200
250
4
2
0
−2
−4
−6

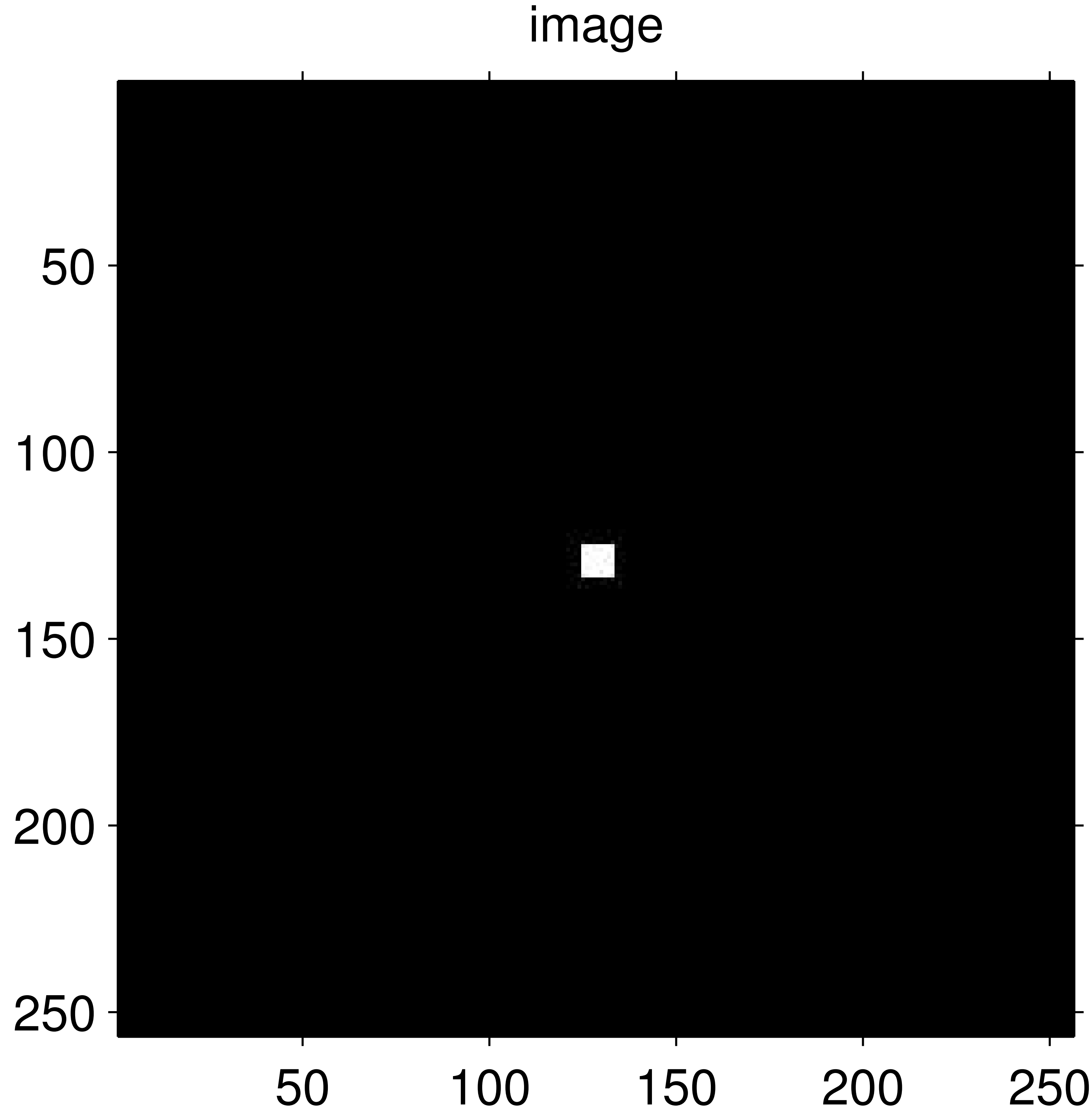
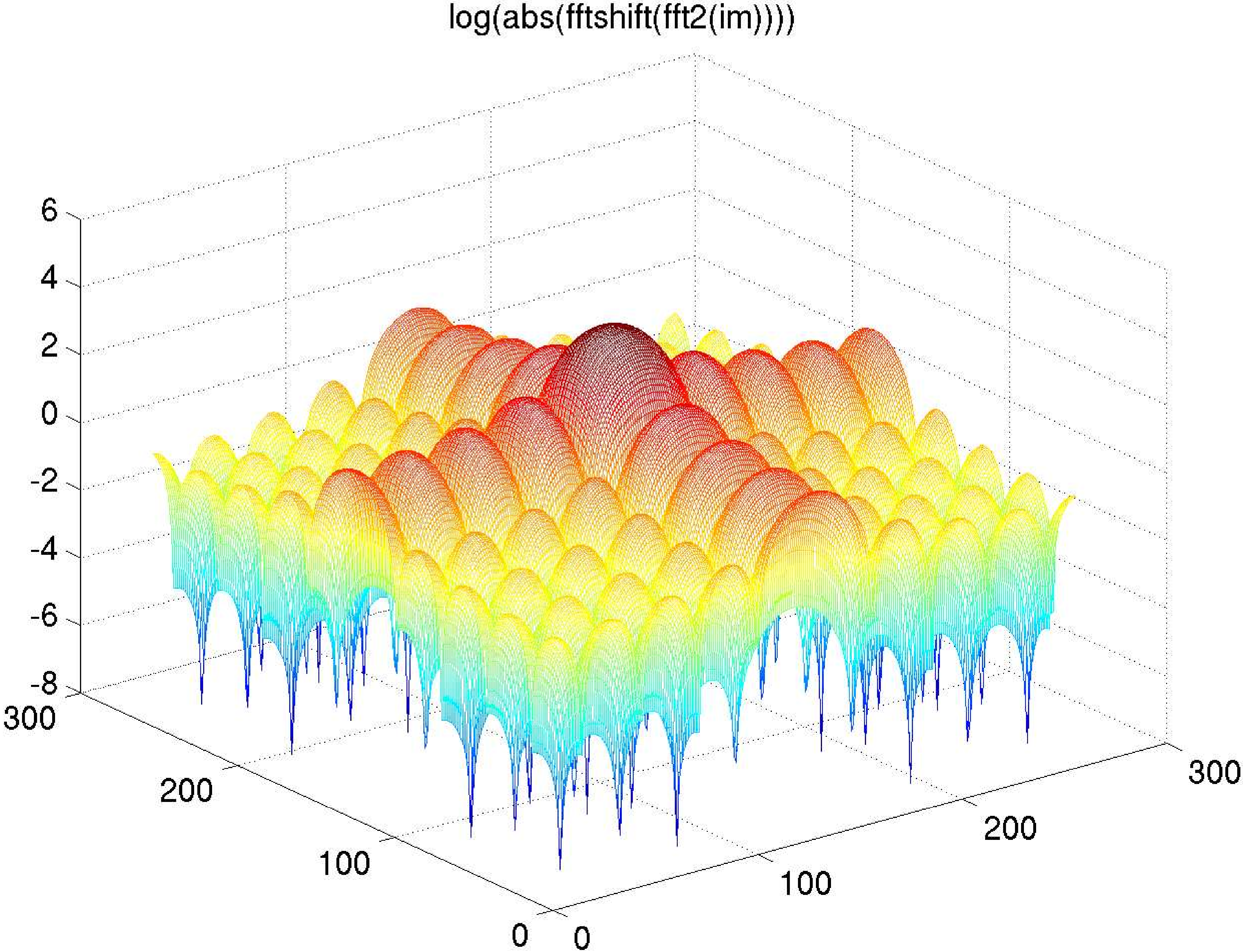
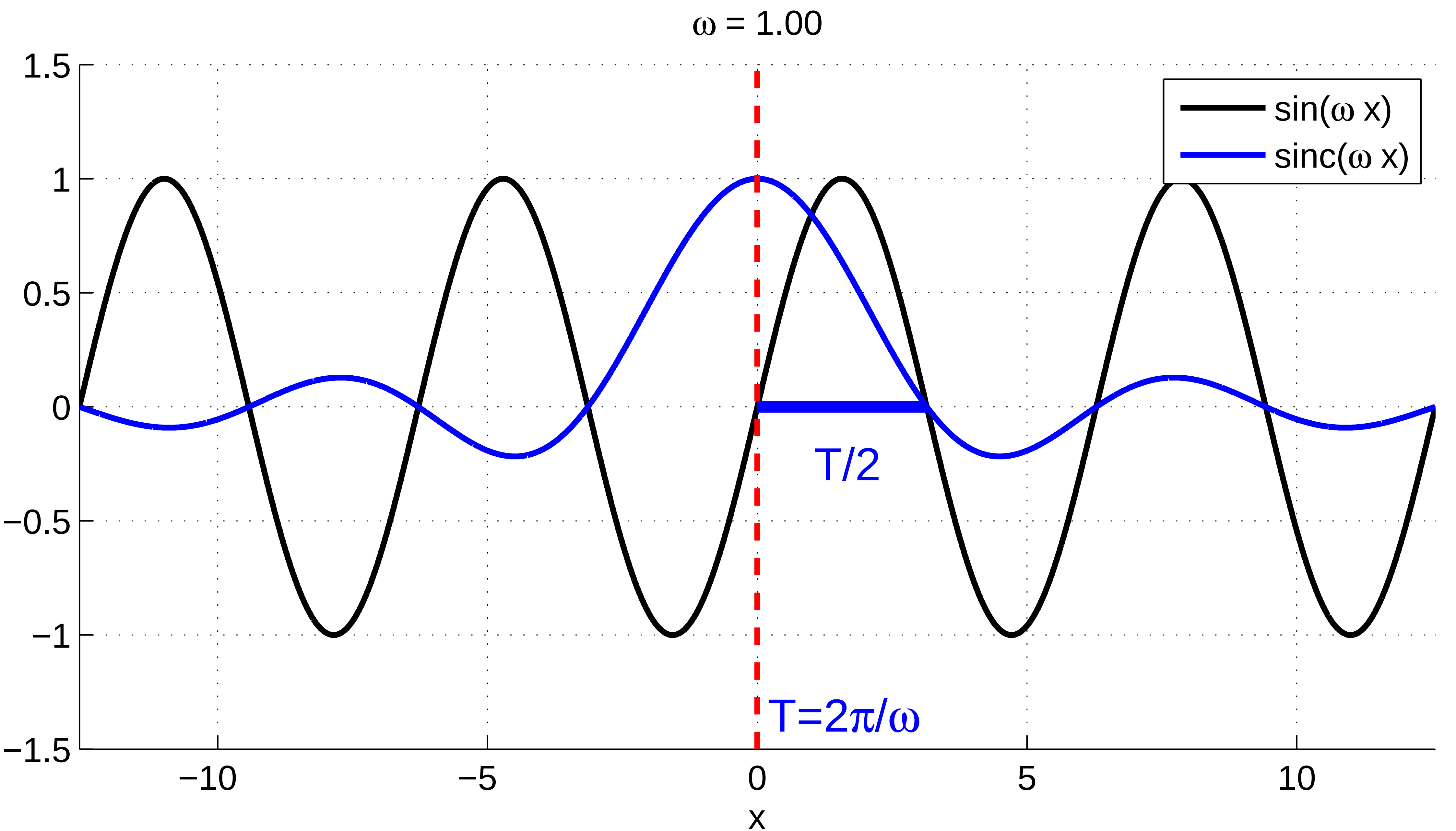
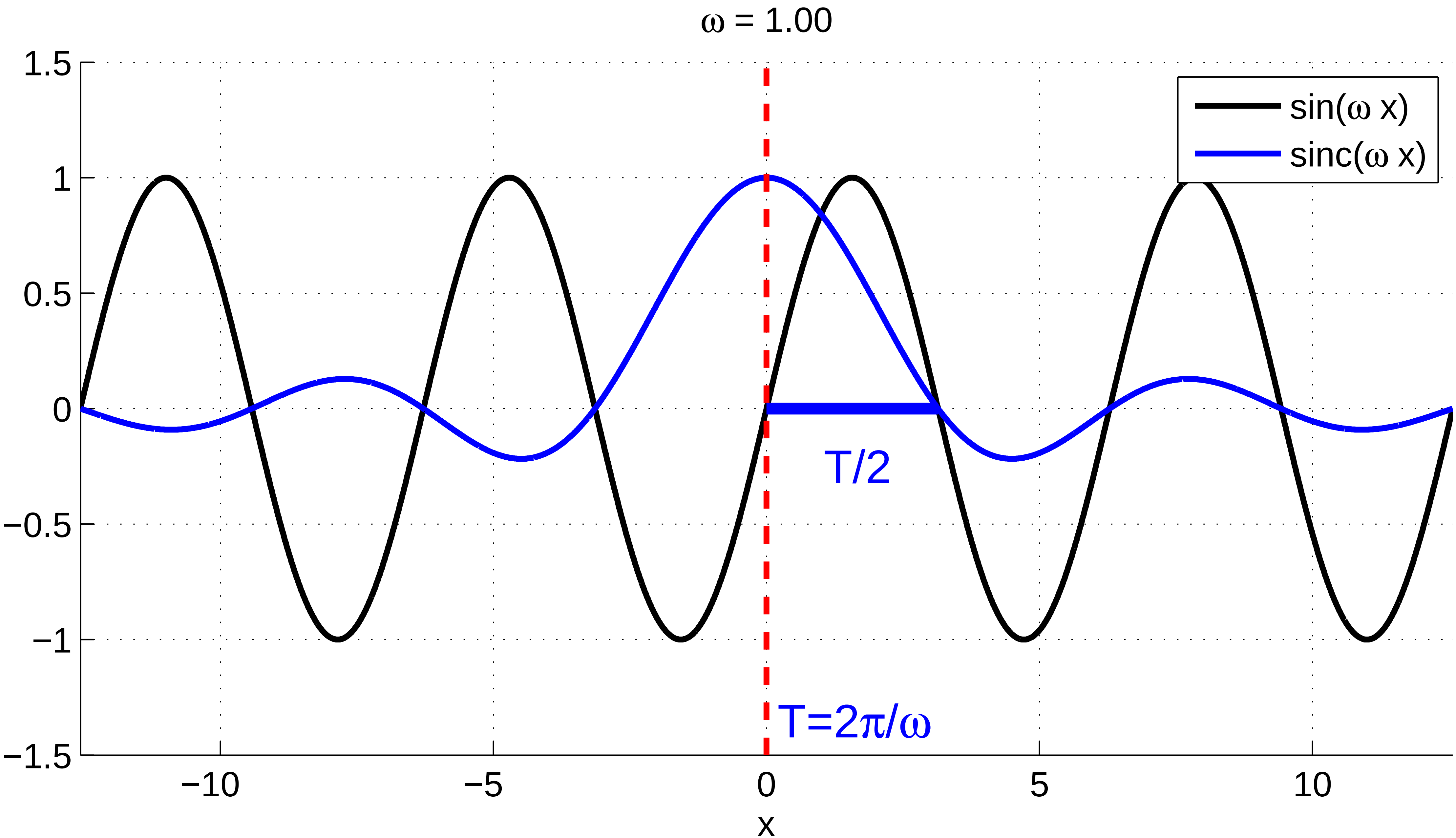


image
50
100
150
200
250
50
100
150
200
250

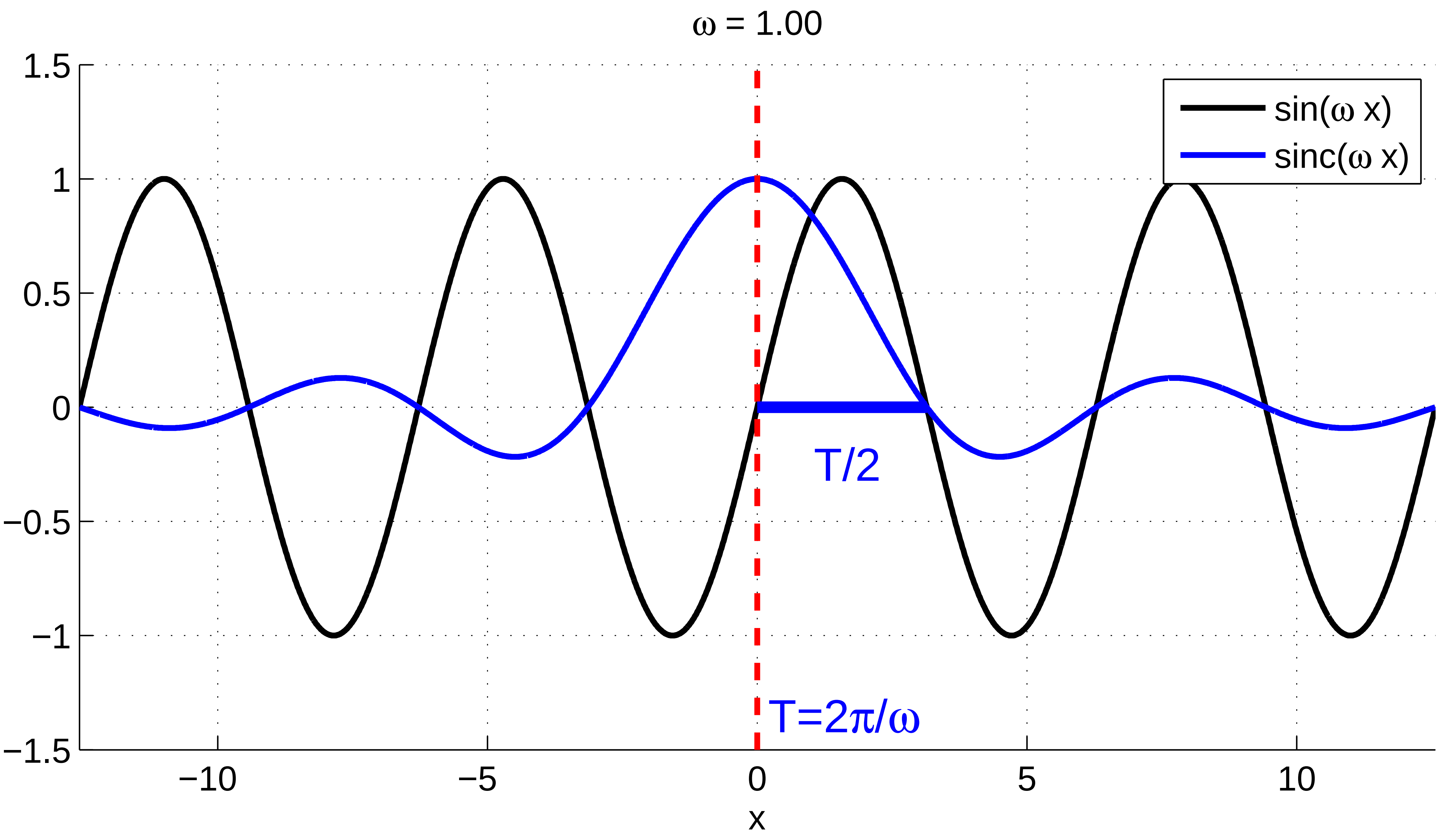

log(abs(fftshift(fft2(im))))
6
4
2
0
-2
-4
-6
-8
300
200
100
0
0
100
200
300



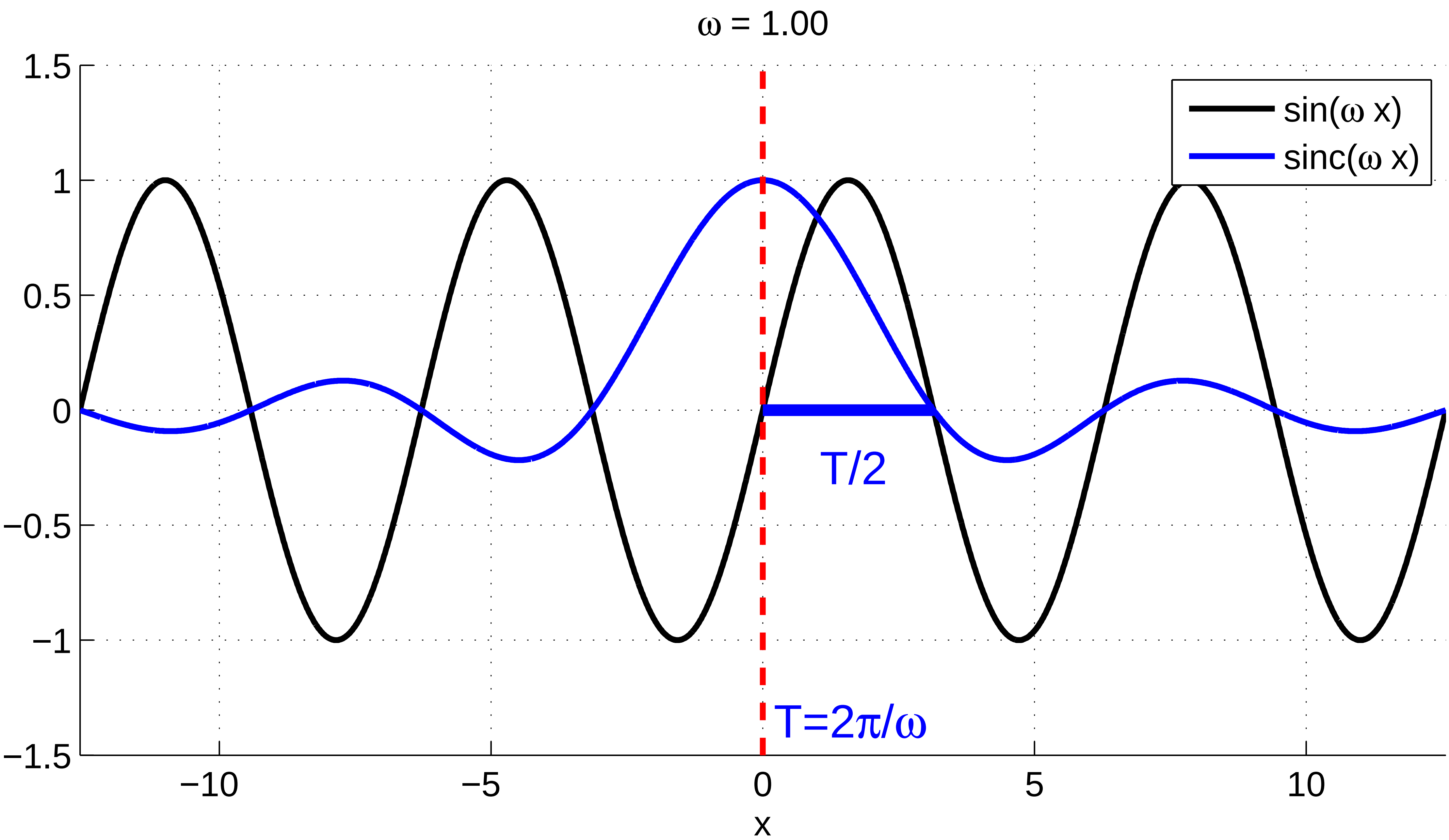
ω = 1.00
1.5
1
0.5
0
−0.5
−1
−1.5
−10
−5
0
5
10
x
sin(ω x)
sinc(ω x)
T/2
T=2π/ω

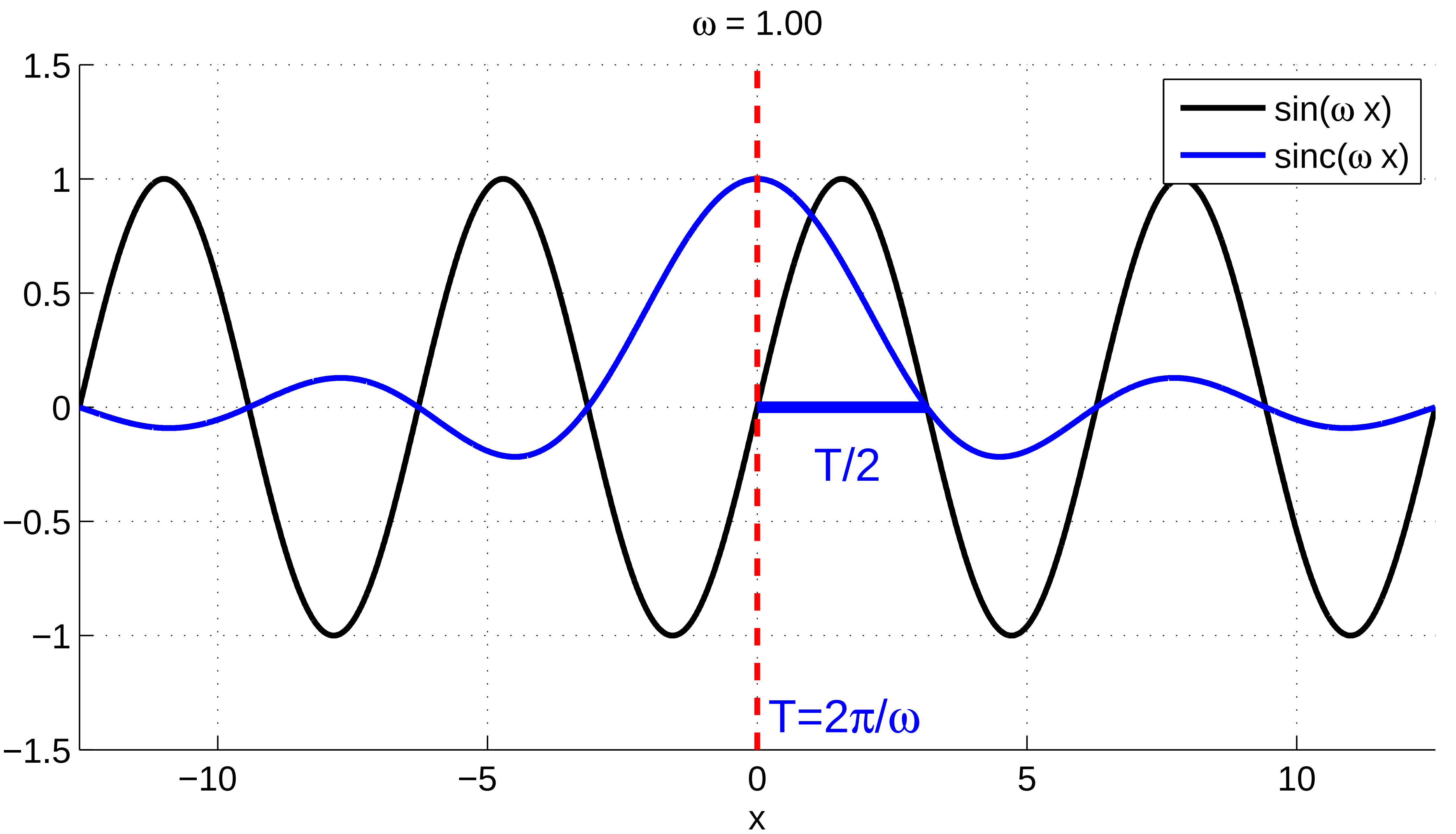


ω = 1.00
1.5
1
0.5
0
−0.5
−1
−1.5
−10
−5
0
5
10
x
sin(ω x)
sinc(ω x)
T/2
T=2π/ω

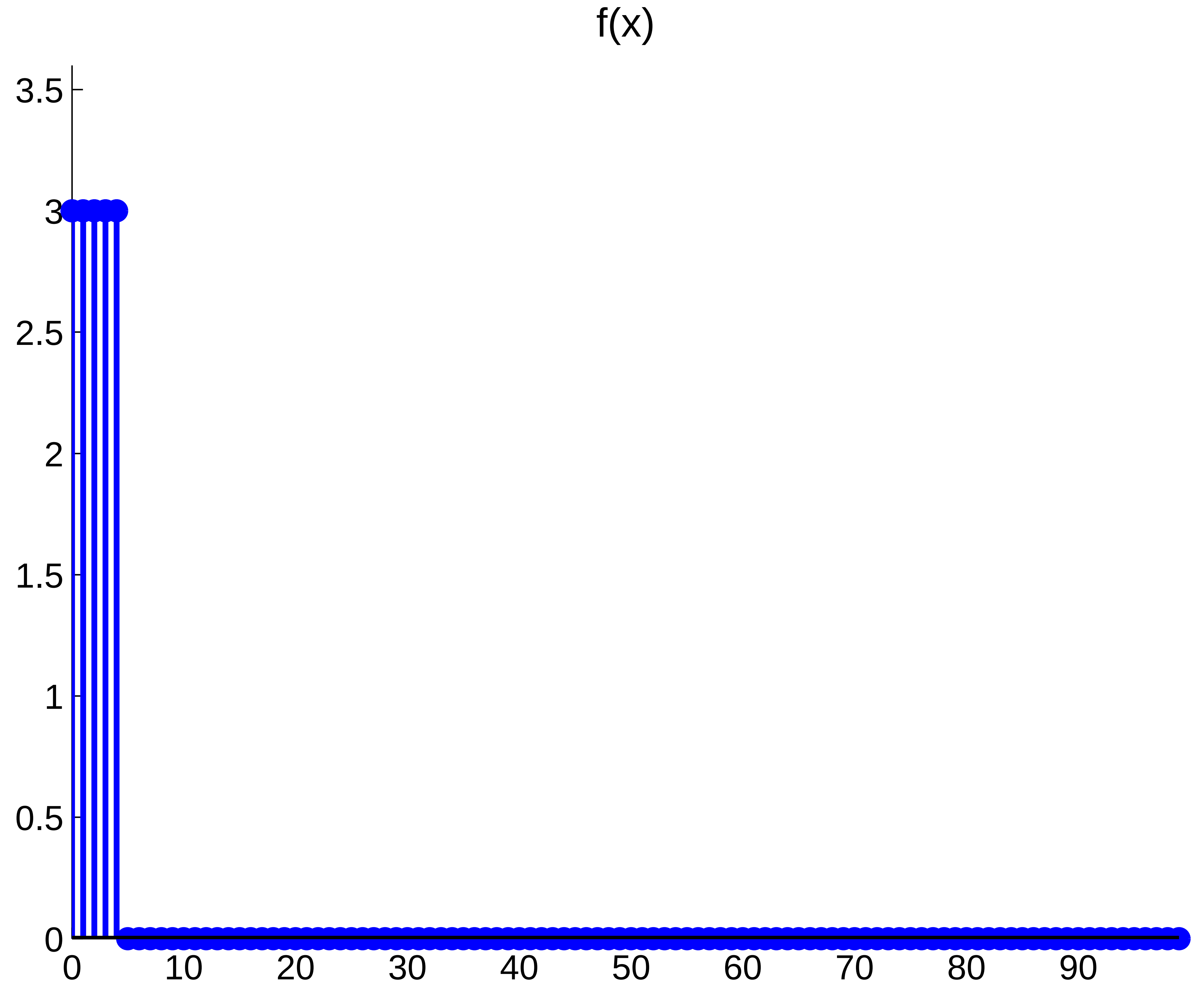


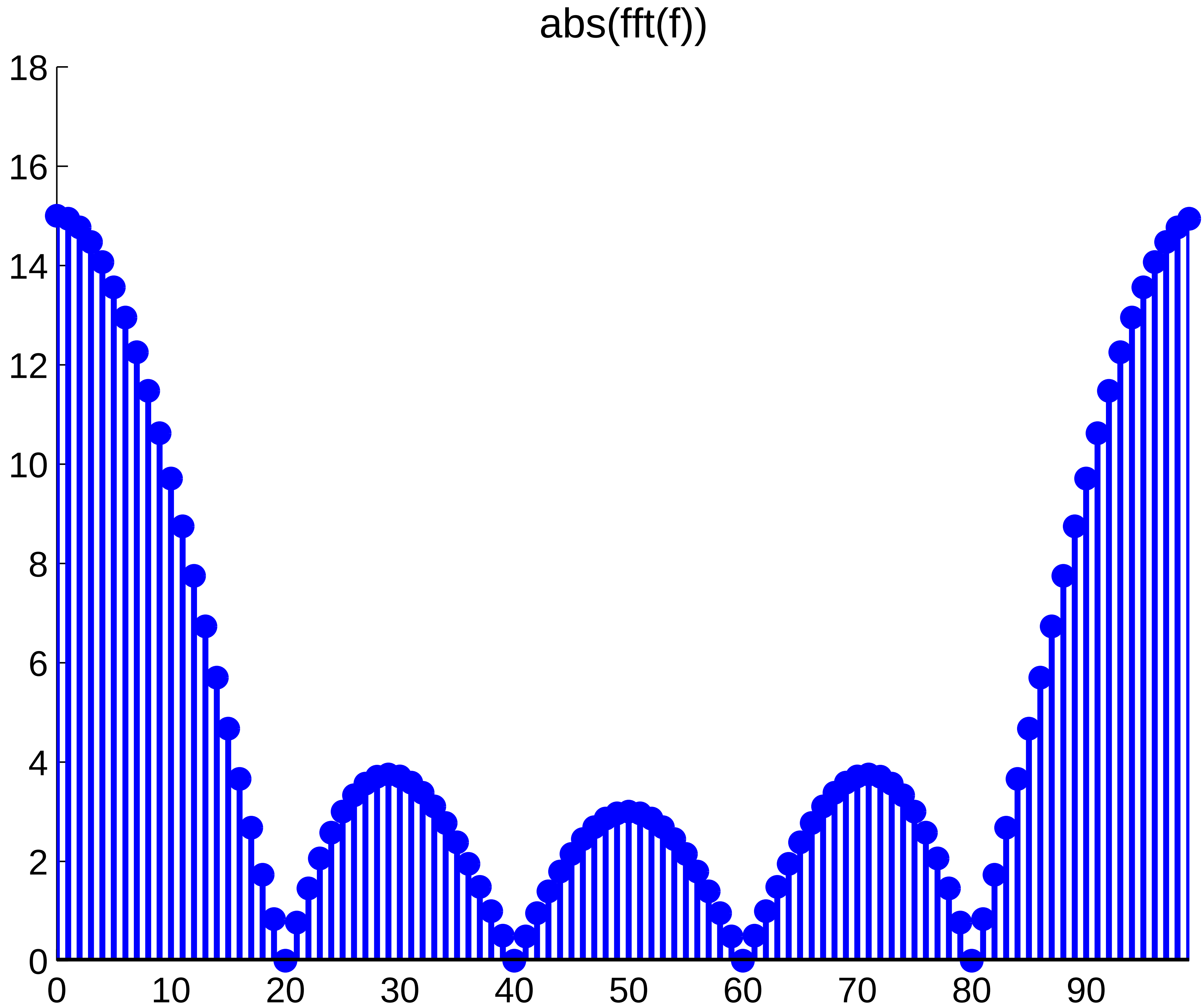
ω = 1.00
1.5
1
0.5
0
−0.5
−1
−1.5
−10
−5
0
5
10
x
sin(ω x)
sinc(ω x)
T/2
T=2π/ω


ω = 1.00
sin(ω x)
sinc(ω x)
T/2
T=2π/ω
1.5
1
0.5
0
−0.5
−1
−1.5
−10
−5
0
5
10
x

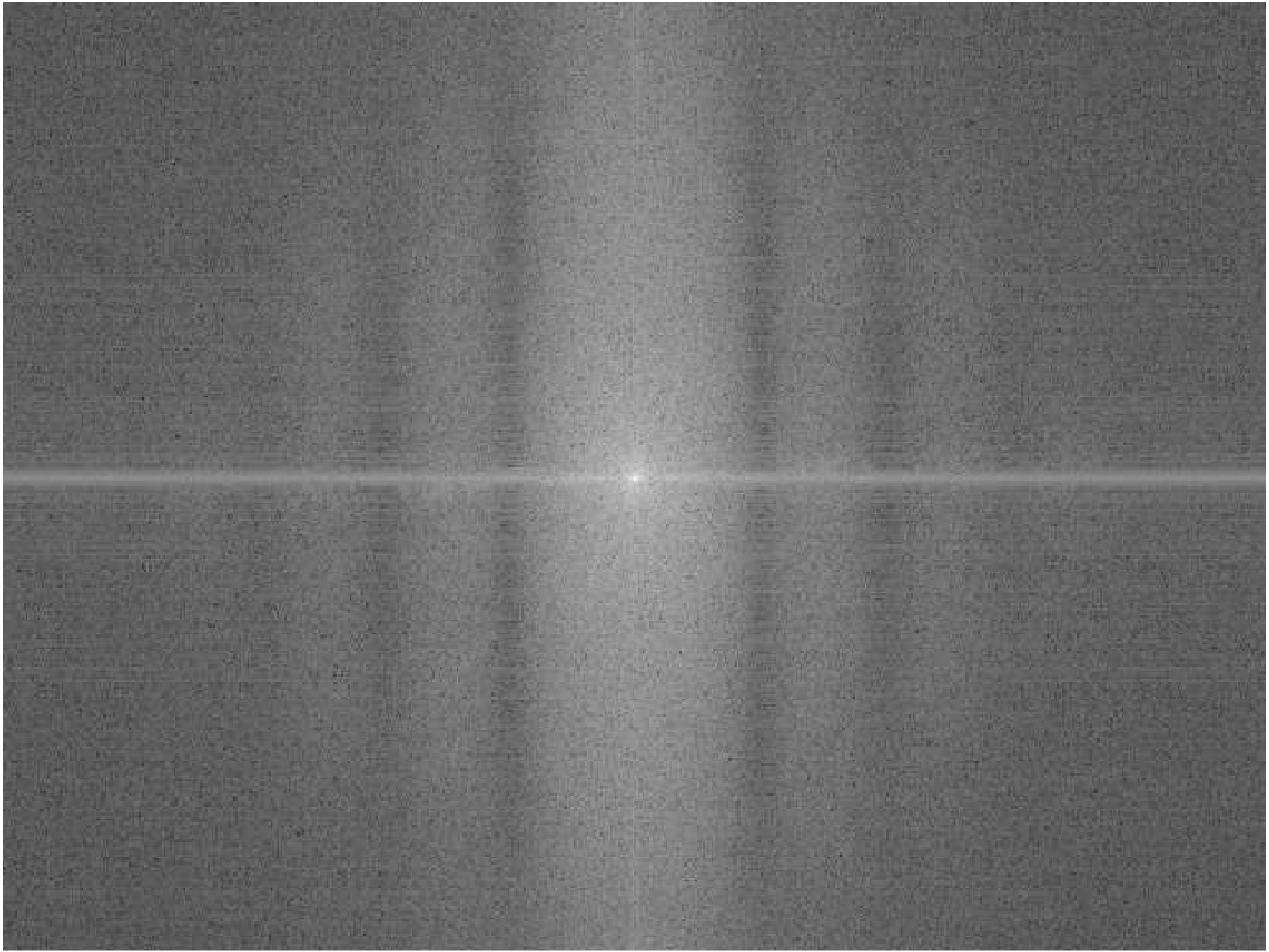


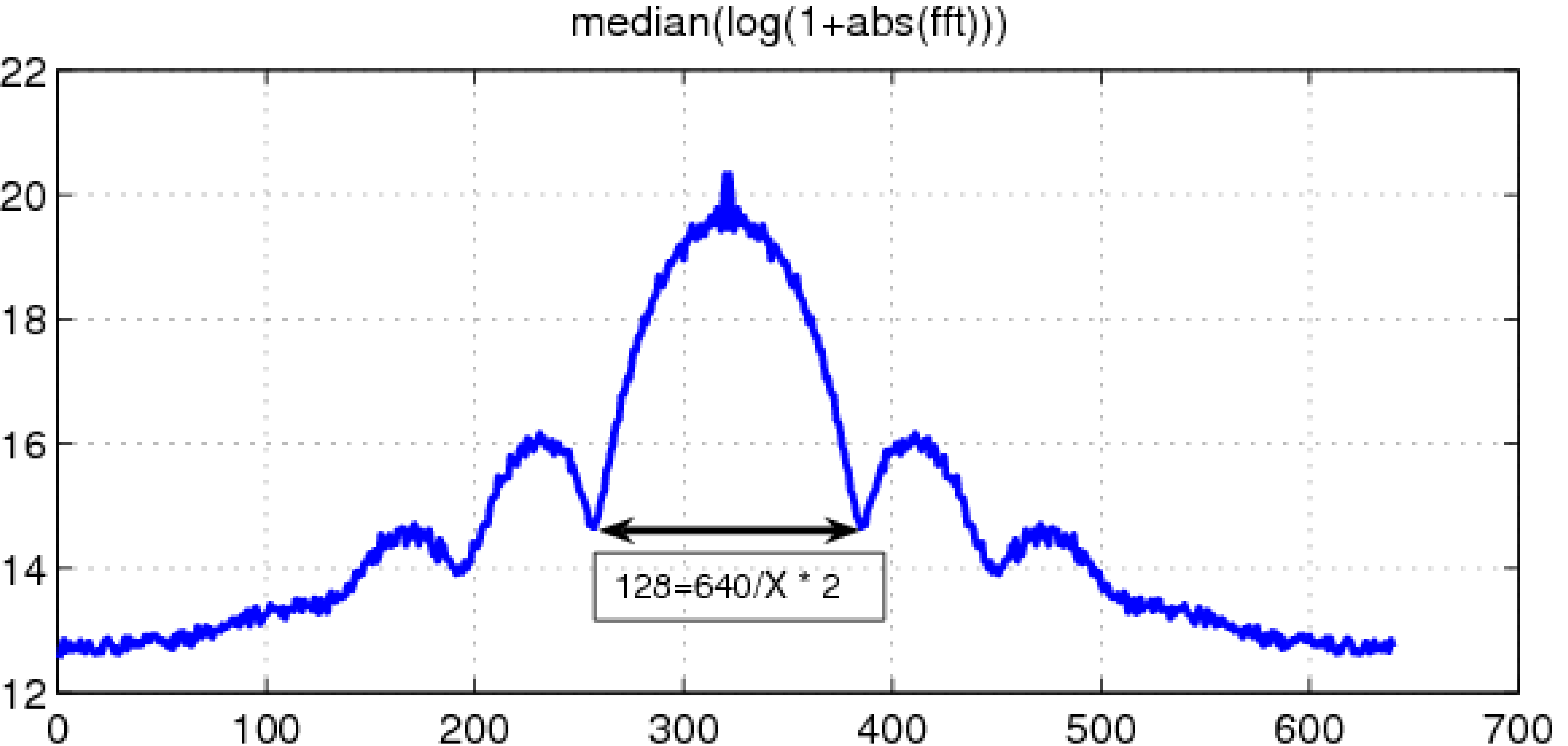
ω = 1.00
sin(ω x)
sinc(ω x)
T/2
T=2π/ω
1.5
1
0.5
0
−0.5
−1
−1.5
−10
−5
0
5
10
x


f(x)
3.5
3
2.5
2
1.5
1
0.5
0
0
10
20
30
40
50
60
70
80
90


abs(fft(f))
0
2
4
6
8
10
12
14
16
18
0
10
20
30
40
50
60
70
80
90

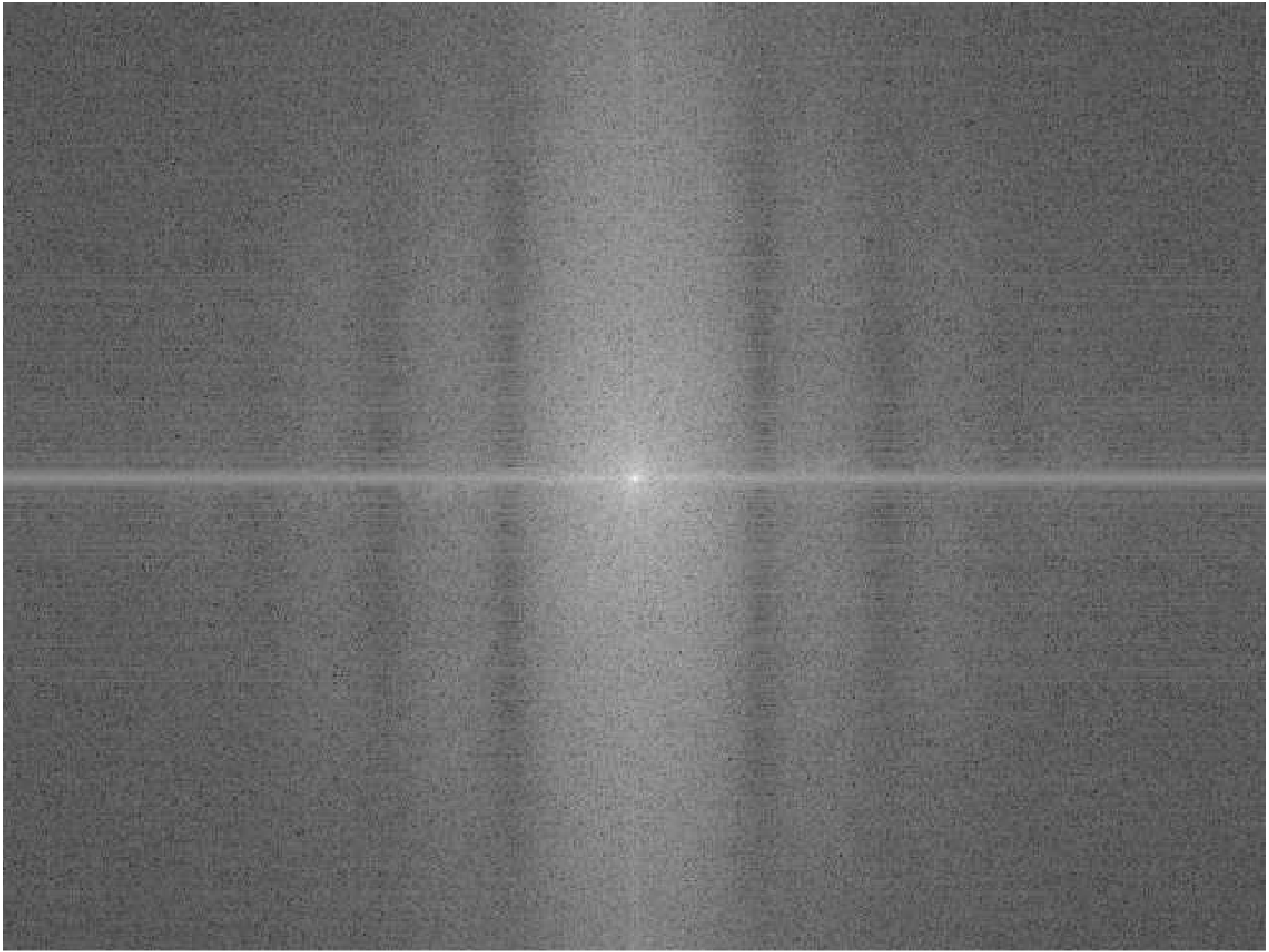
spectrum of blurred image

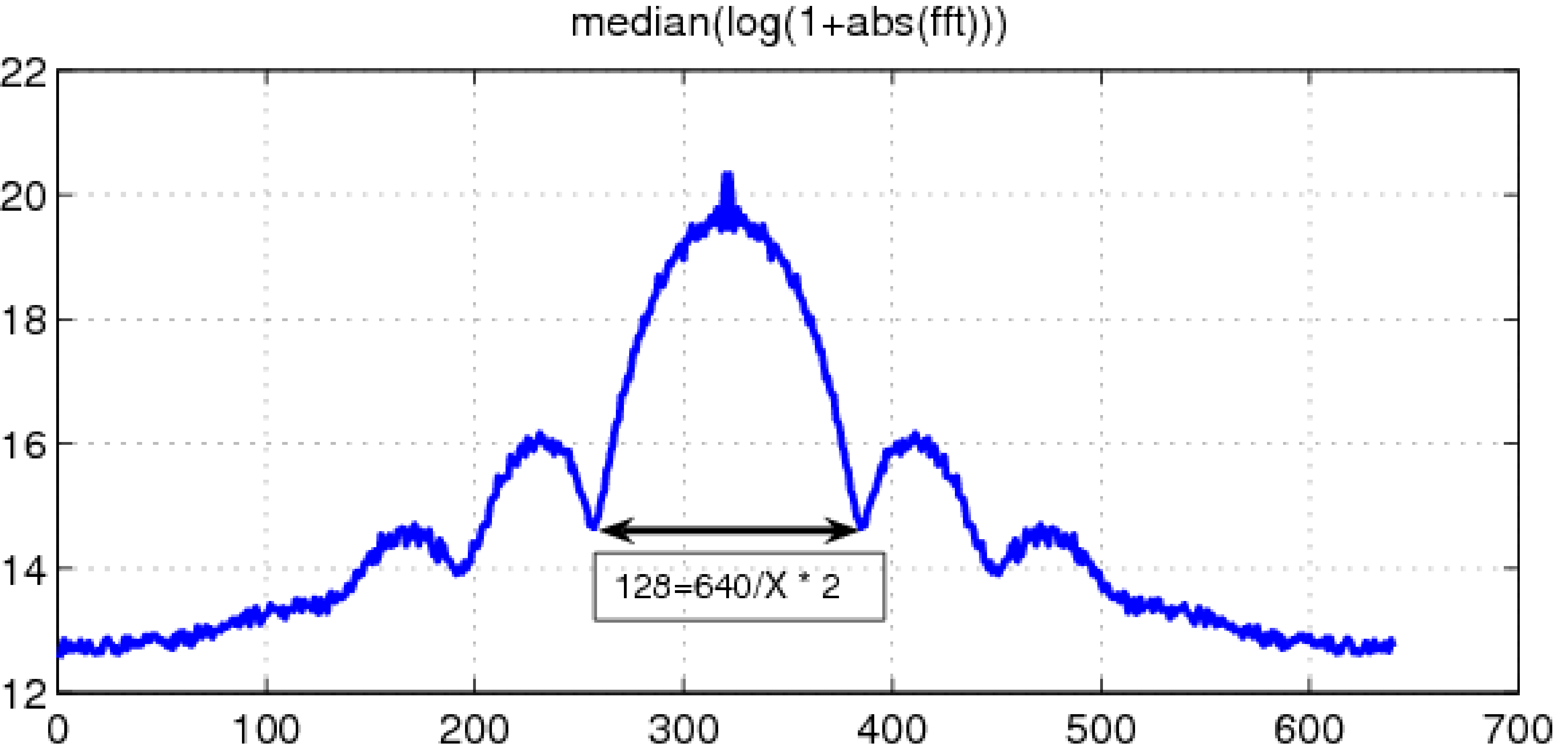




median(log(1+abs(fft)))
128=640/X * 2
22
20
18
16
14
12
0
100
200
300
400
500
600
700

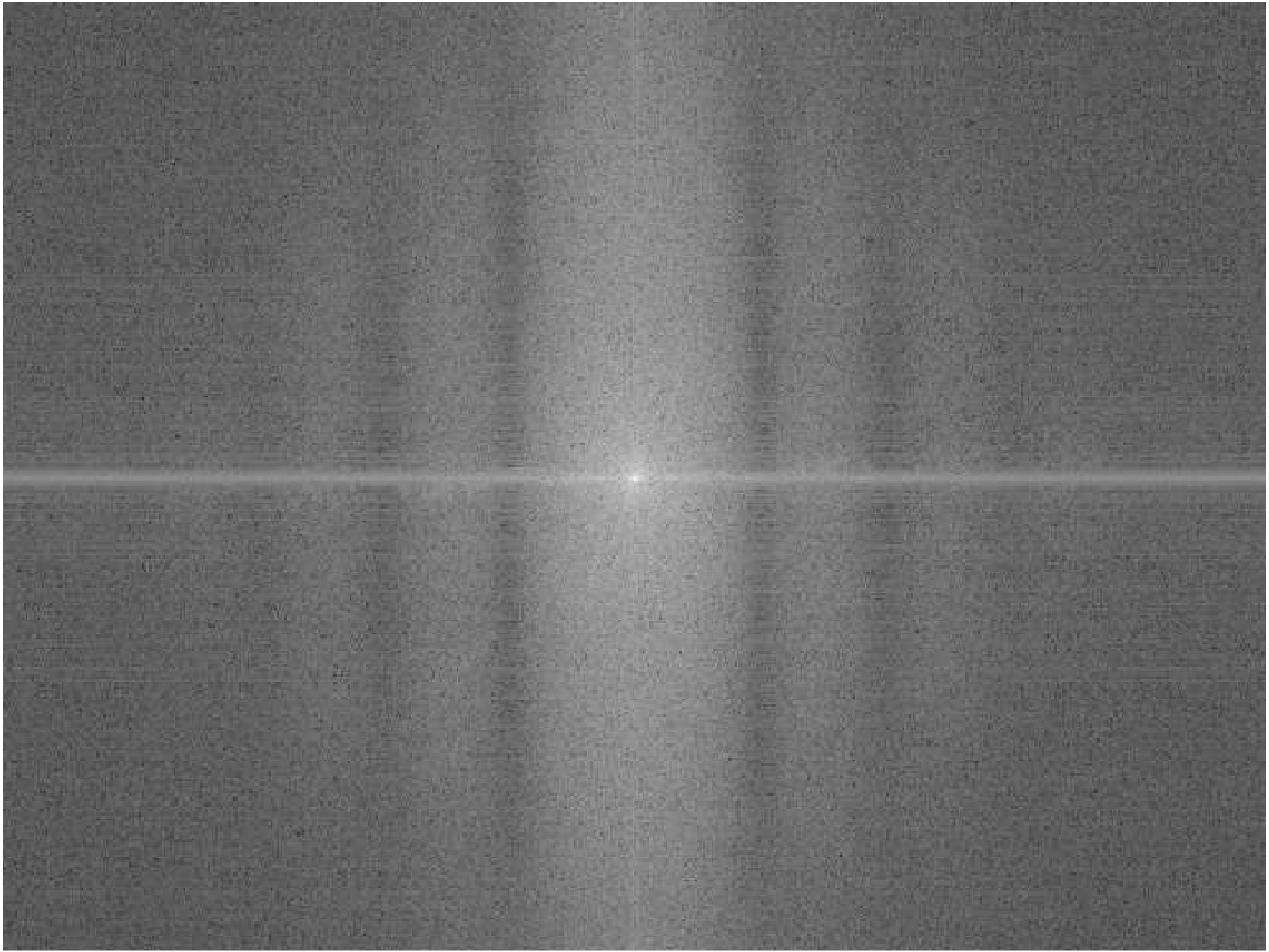
spectrum of blurred image

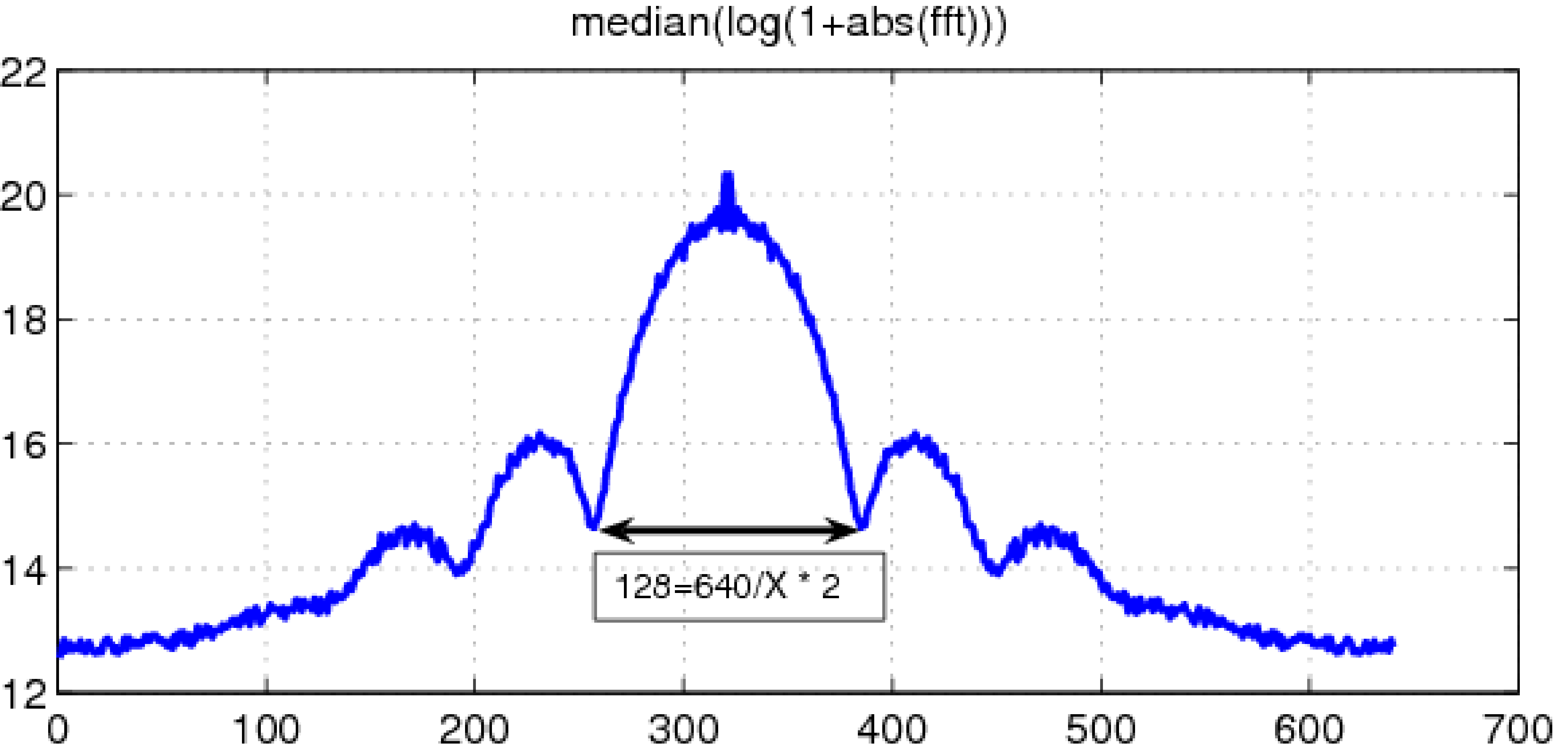




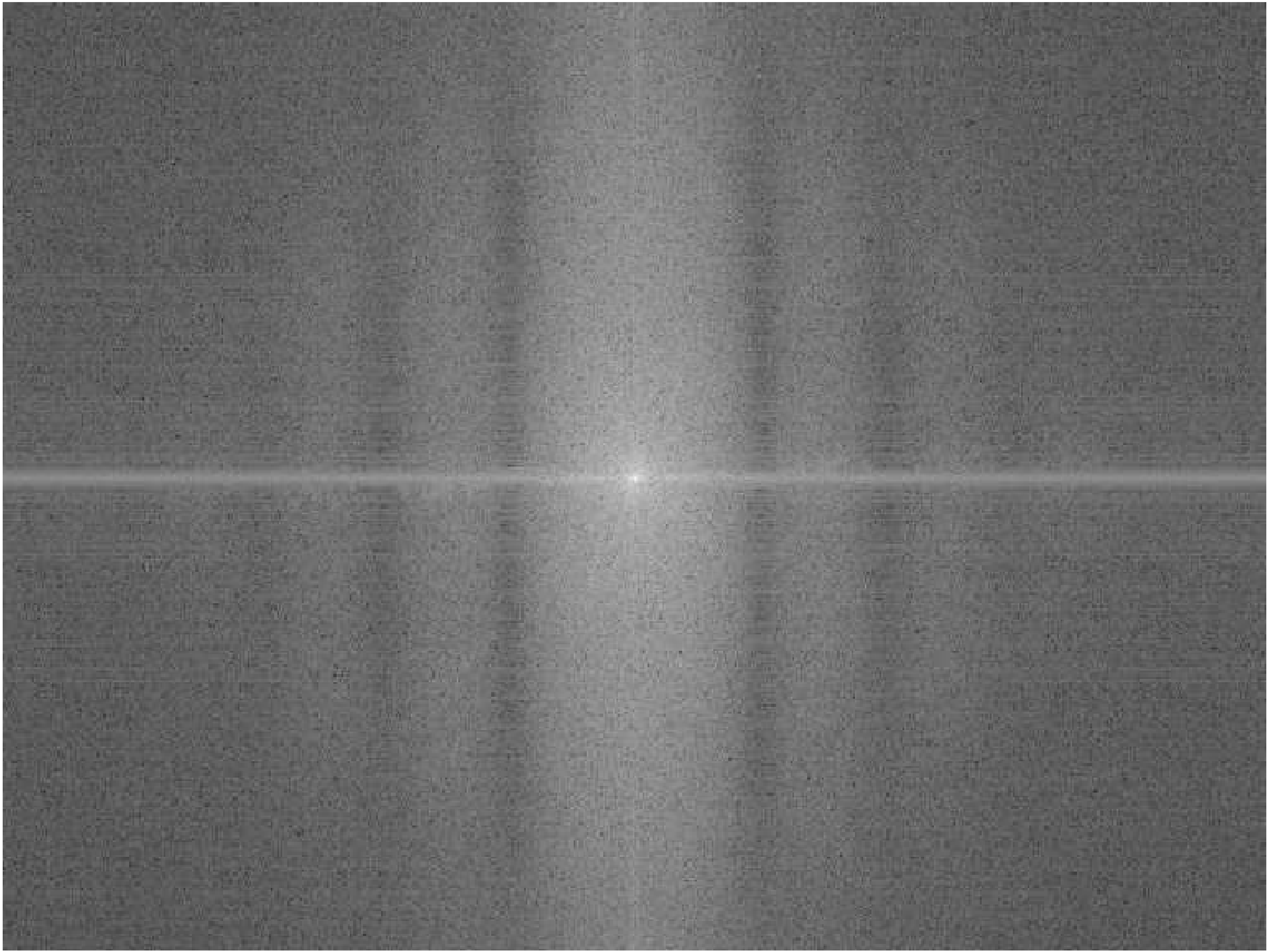
median(log(1+abs(fft)))
22
20
18
16
14
12
0
100
200
300
400
500
600
700
128=640/X * 2

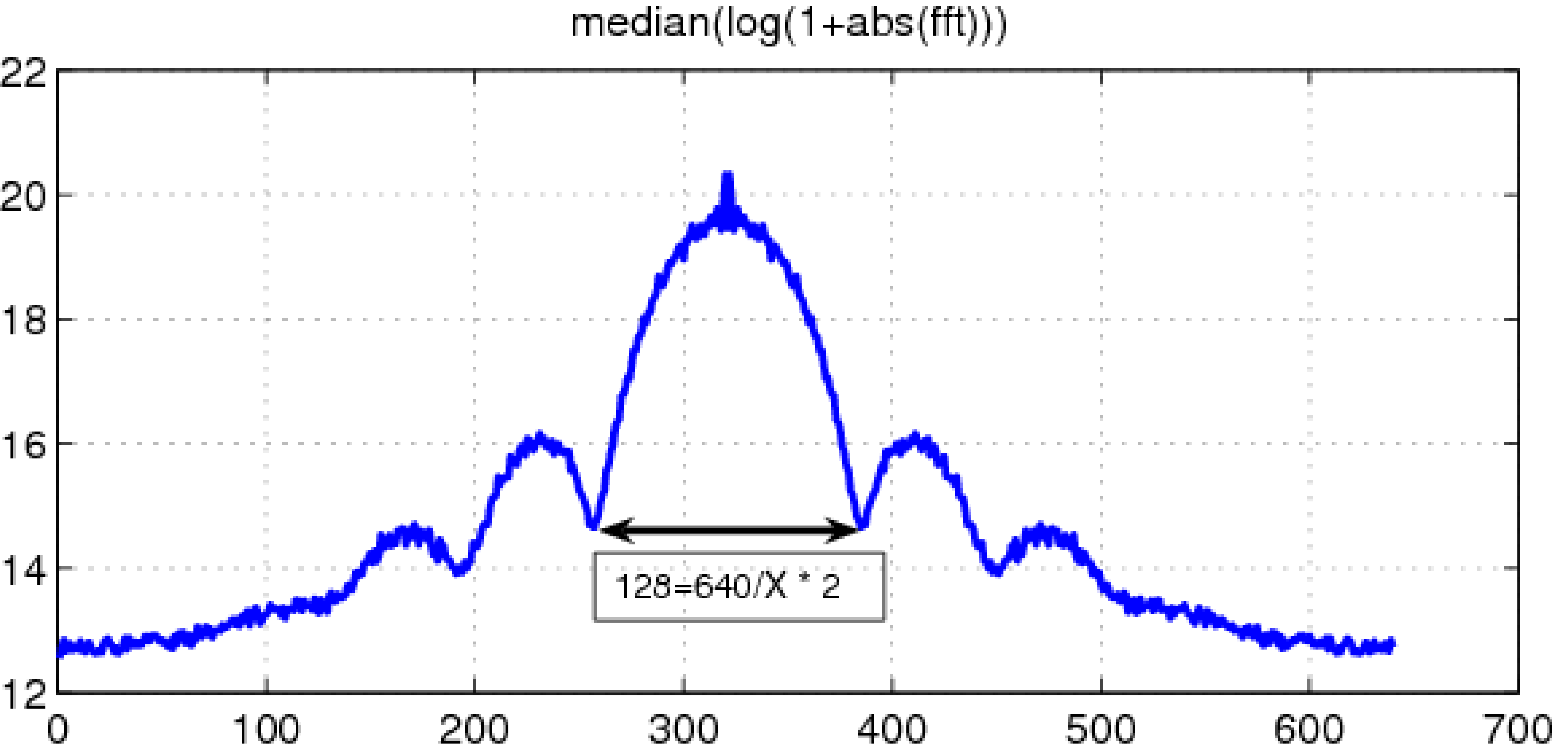
spectrum of blurred image

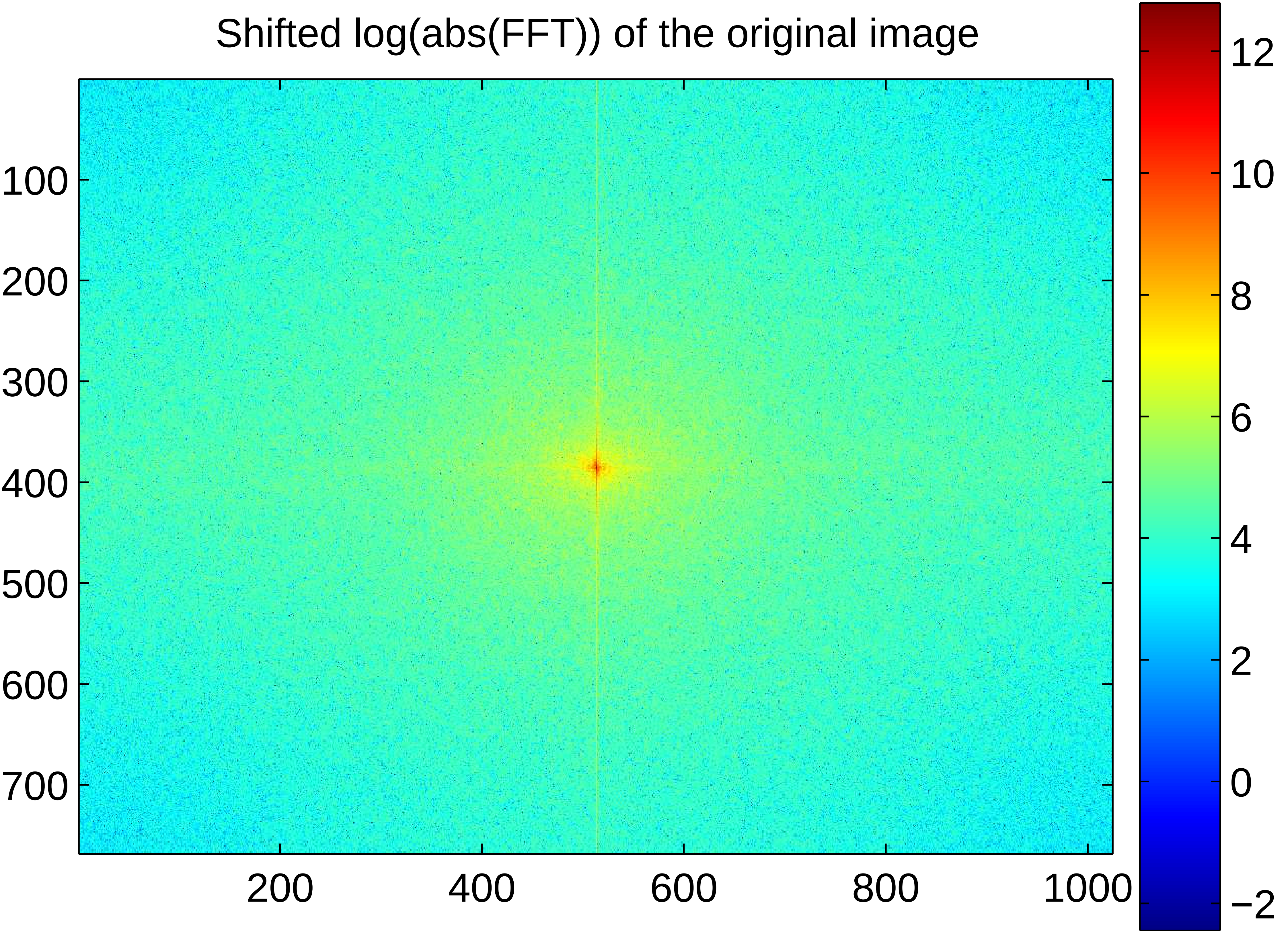


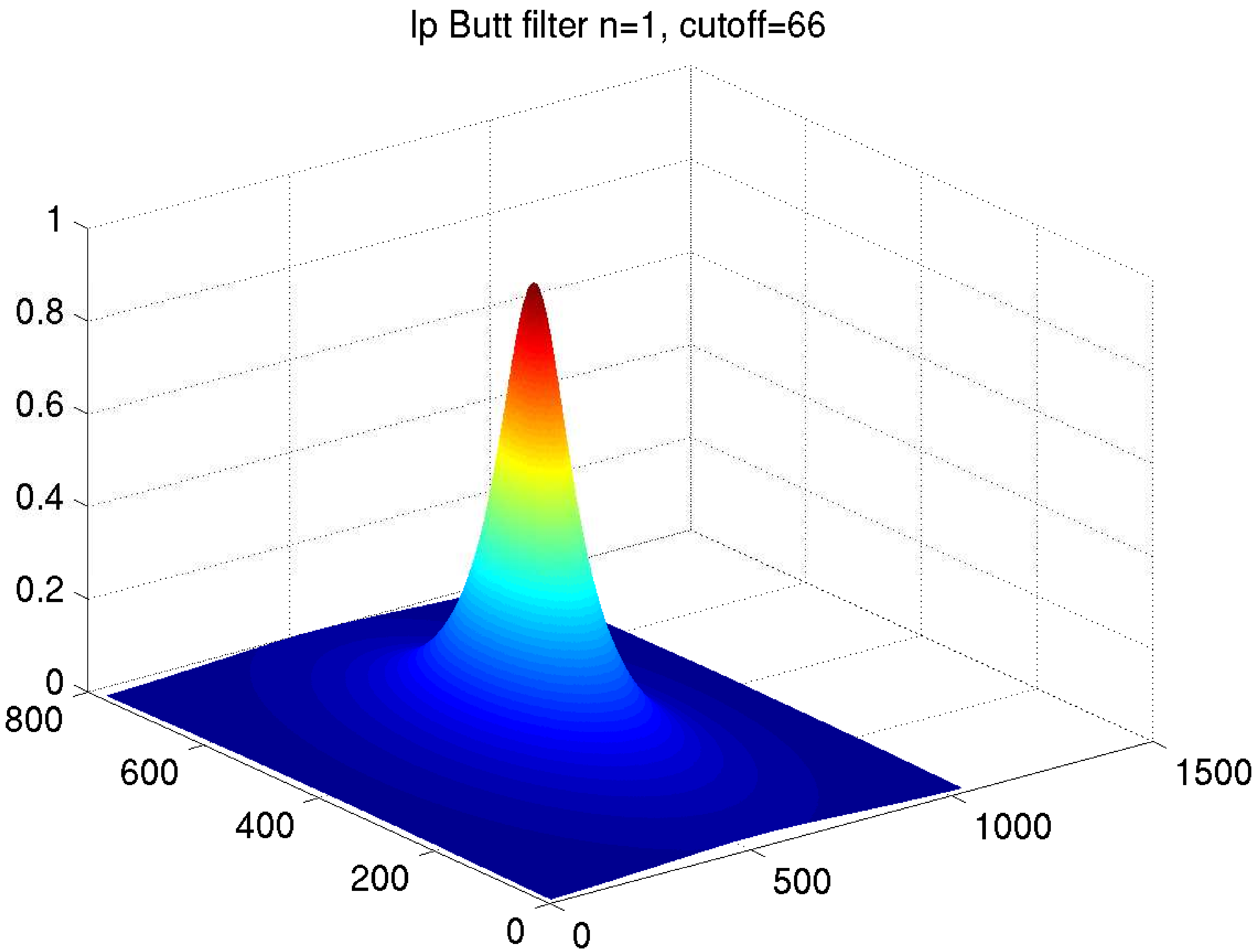

median(log(1+abs(fft)))
22
20
18
16
14
12
0
100
200
300
400
500
600
700
128=640/X * 2

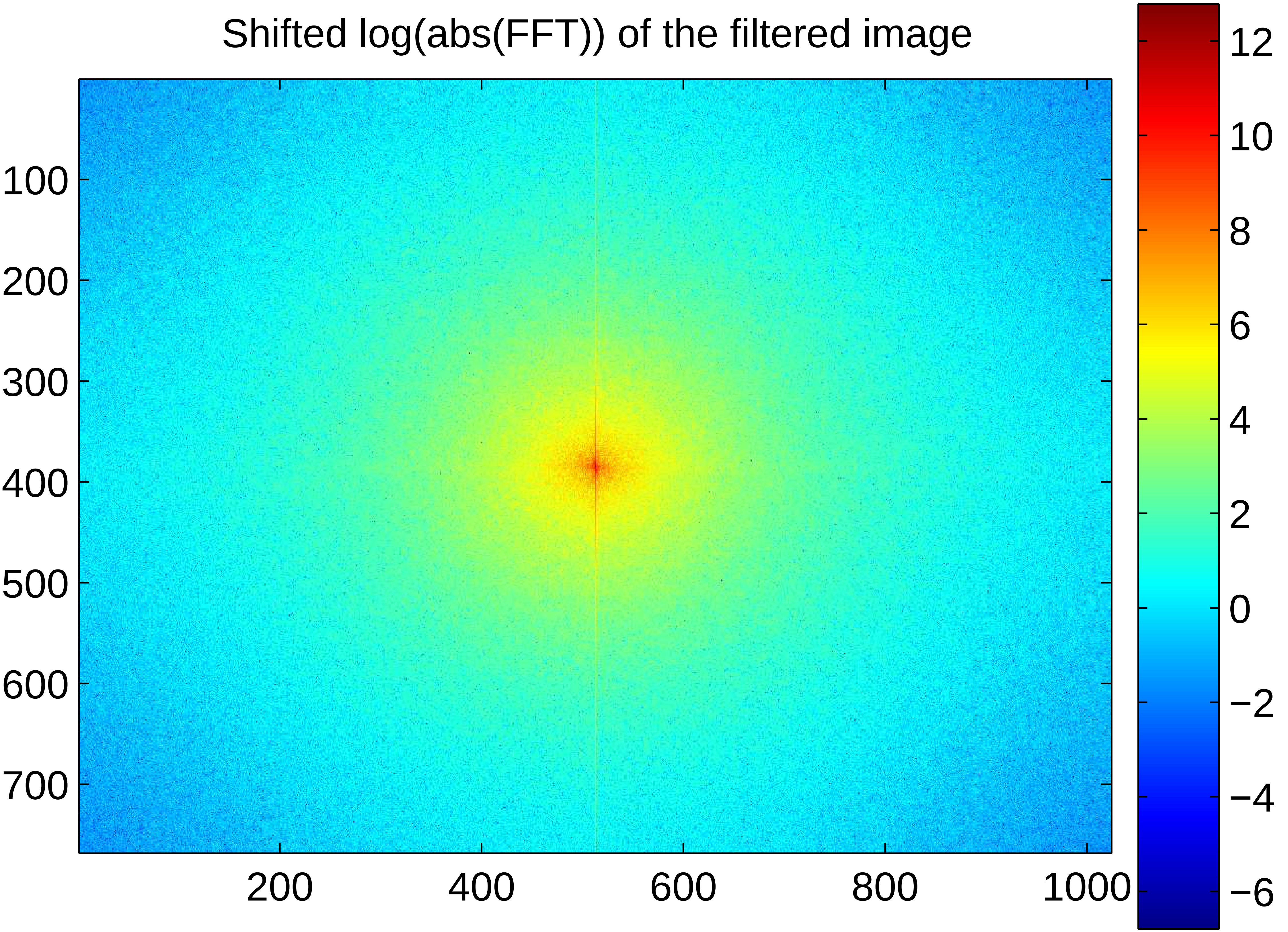
spectrum of blurred image

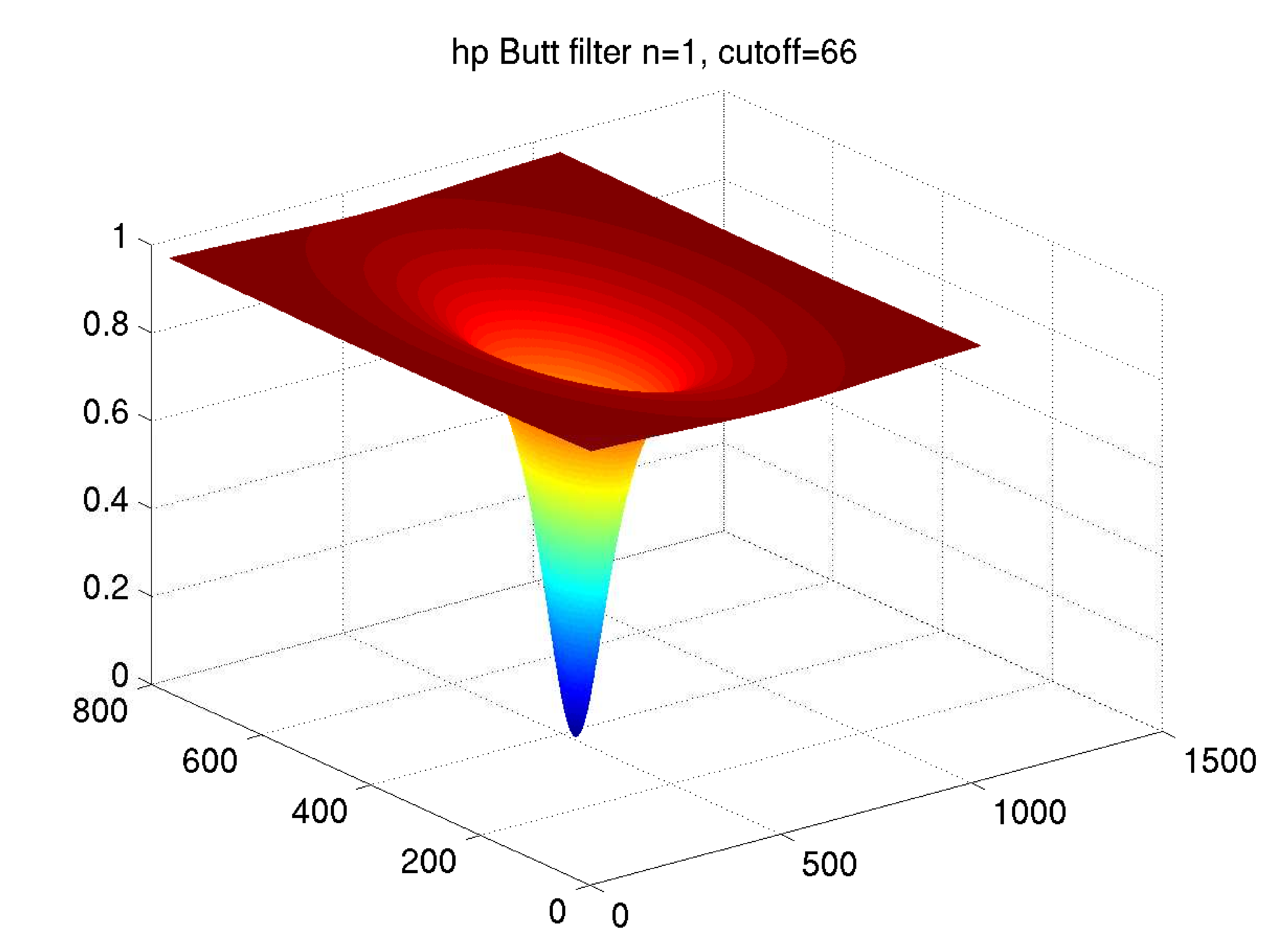


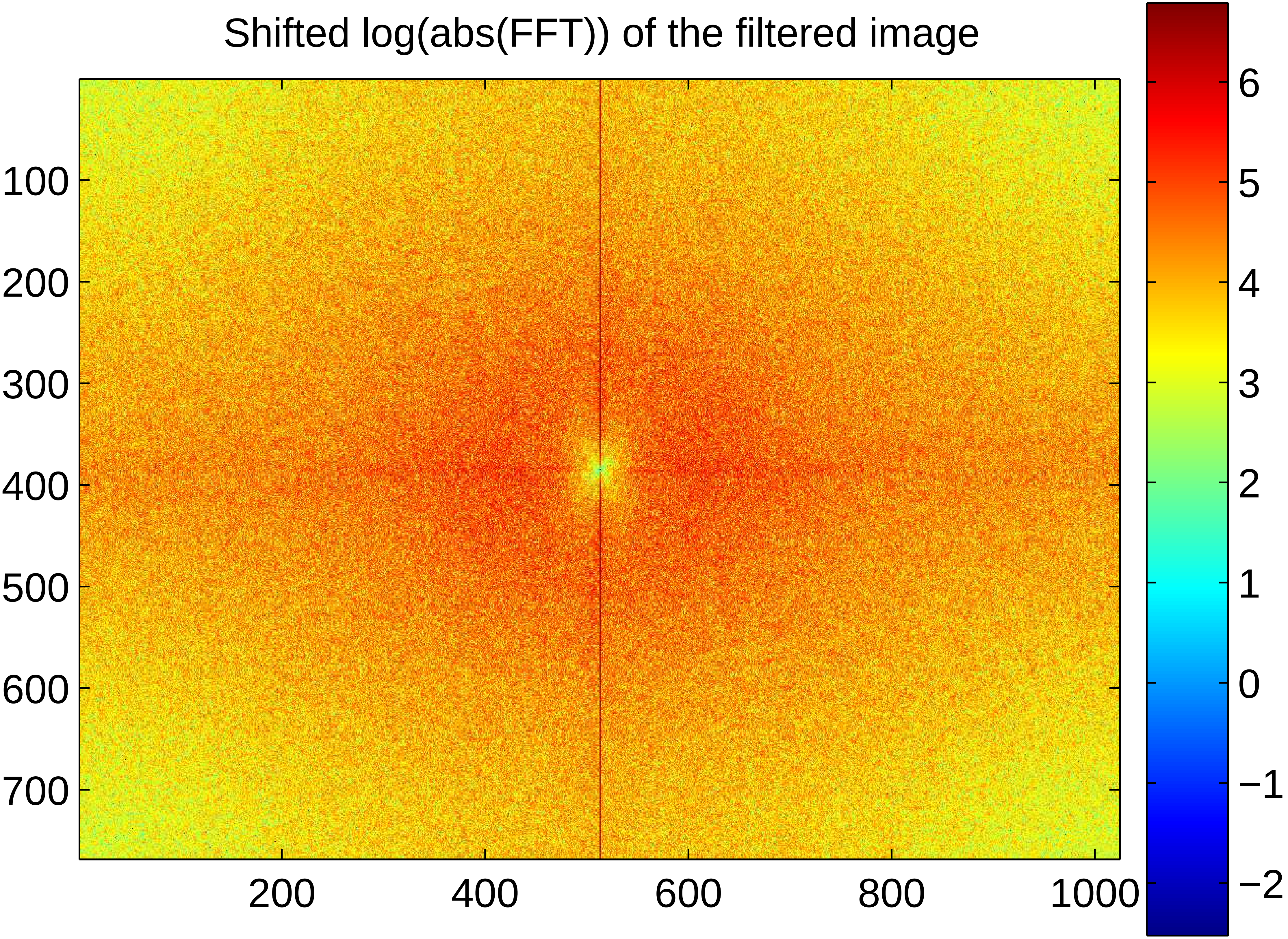

median(log(1+abs(fft)))
128=640/X * 2
22
20
18
16
14
12
0
100
200
300
400
500
600
700

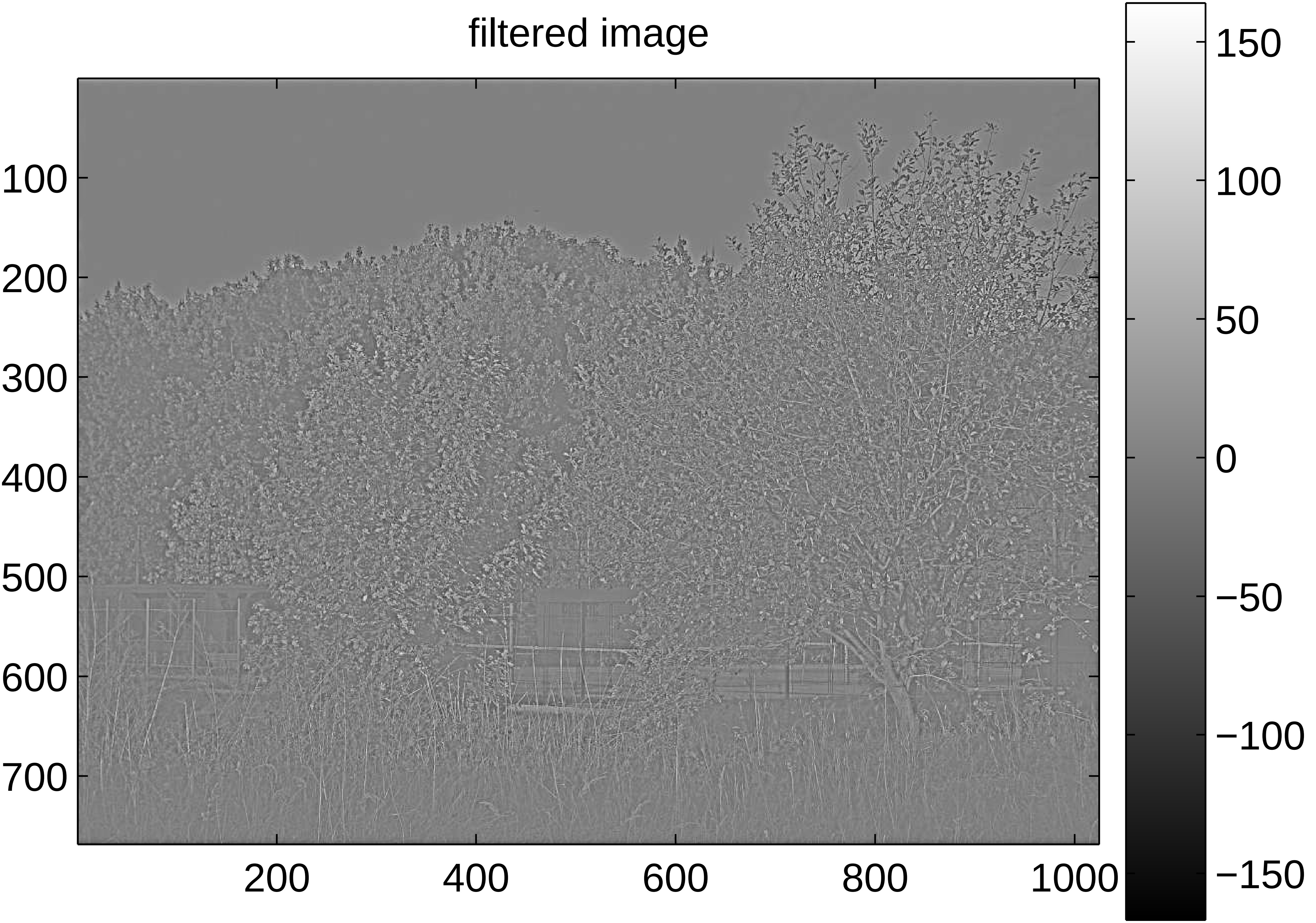

Shifted log(abs(FFT)) of the original image
100
200
300
400
500
600
700
200
400
600
800
1000
12
10
8
6
4
2
0
−2


lp Butt filter n=1, cutoff=66
1
0.8
0.6
0.4
0.2
0
800
600
400
200
0
0
500
1000
1500

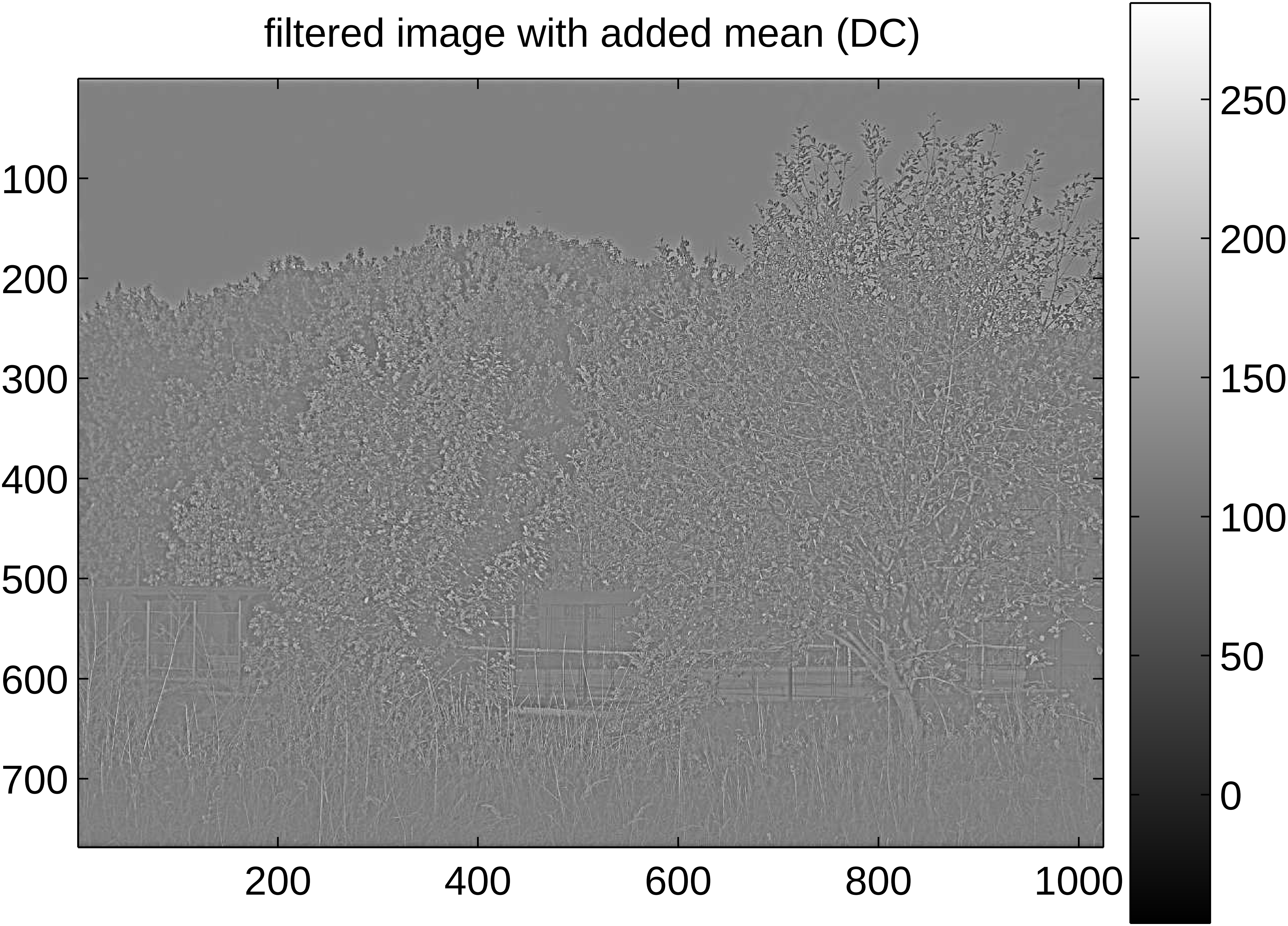

Shifted log(abs(FFT)) of the filtered image
100
200
300
400
500
600
700
200
400
600
800
1000
12
10
8
6
4
2
0
−2
−4
−6

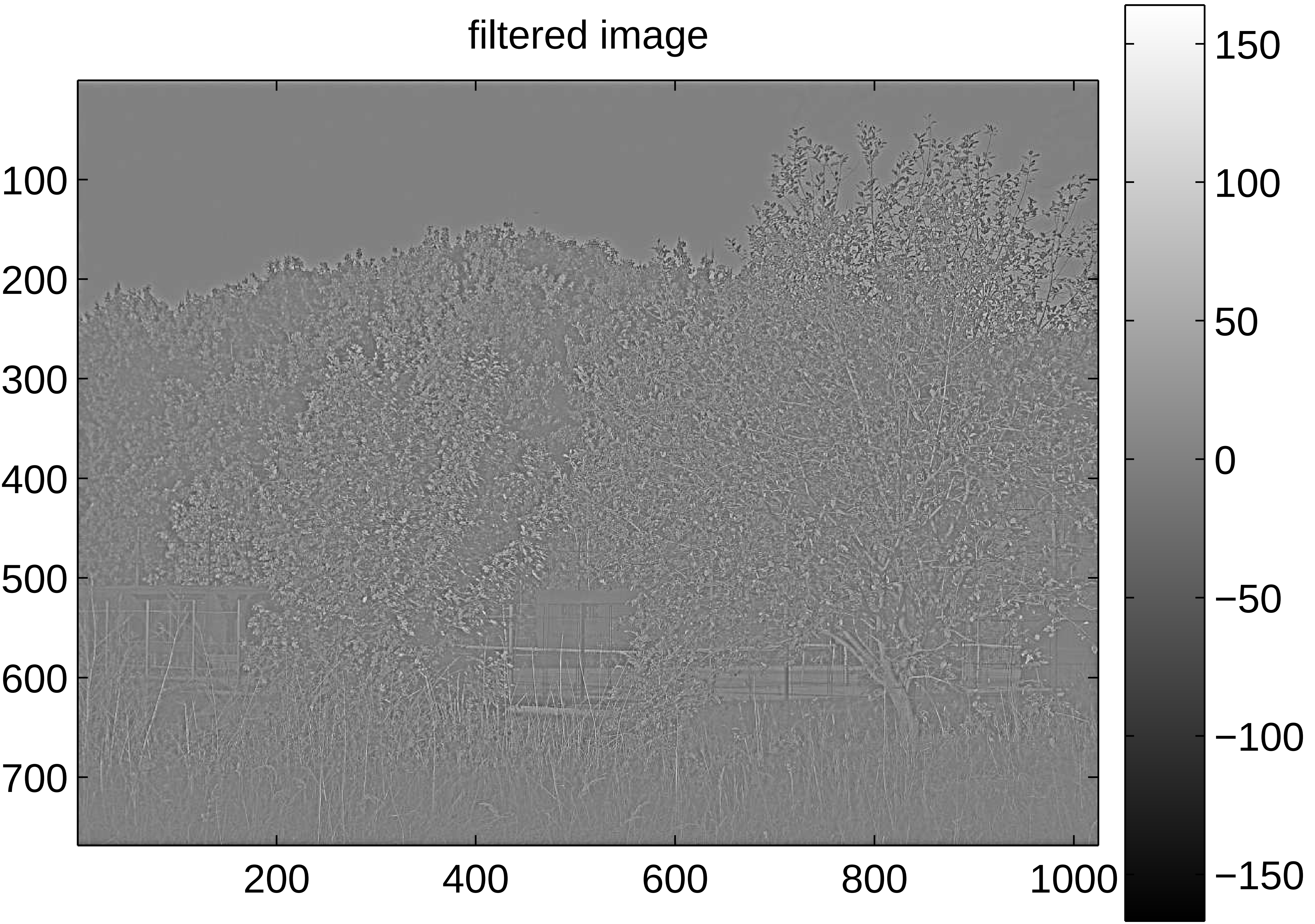

hp Butt filter n=1, cutoff=66
1
0.8
0.6
0.4
0.2
0
800
600
400
200
0
0
500
1000
1500

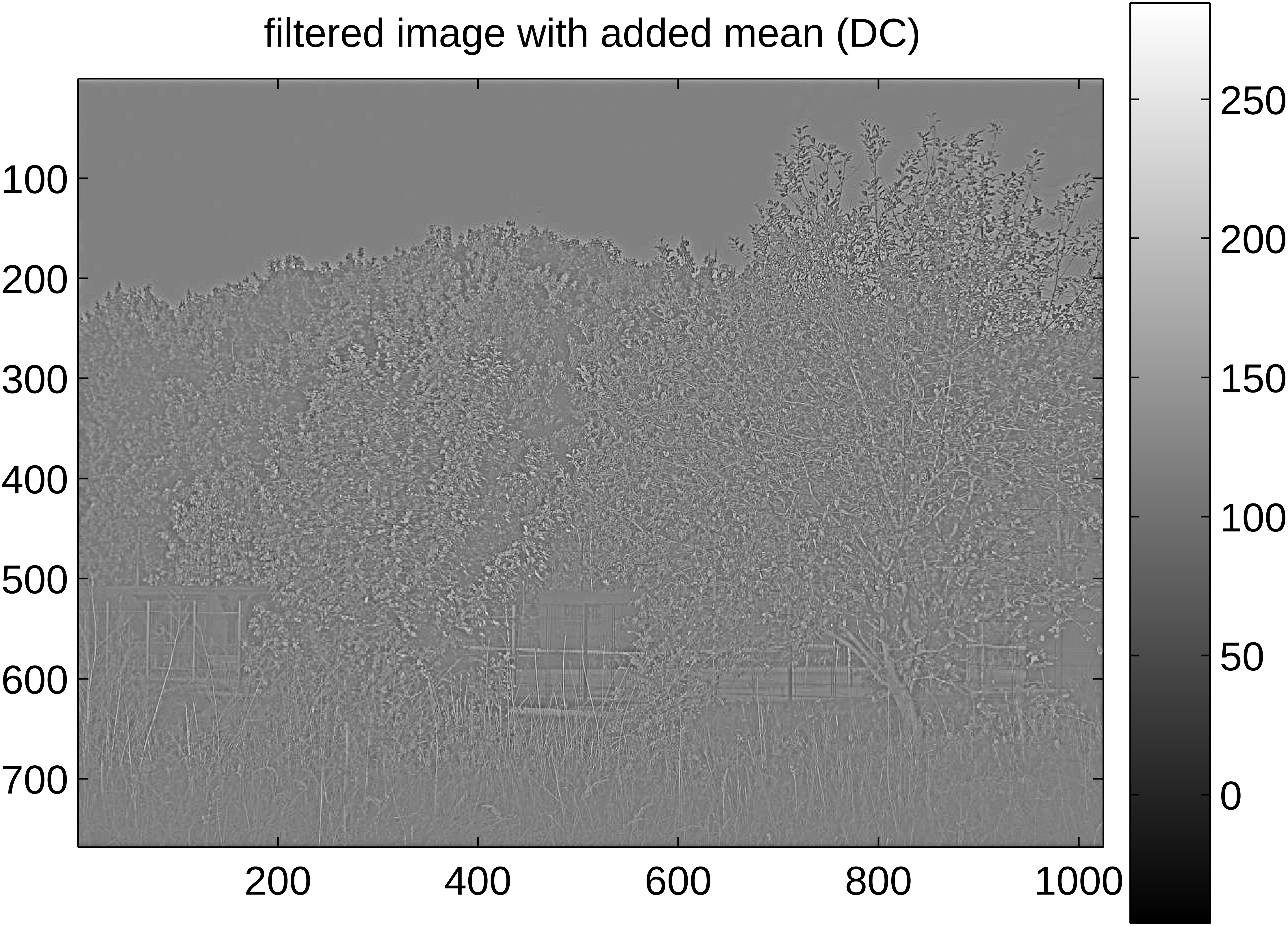

Shifted log(abs(FFT)) of the filtered image
100
200
300
400
500
600
700
200
400
600
800
1000
6
5
4
3
2
1
0
−1
−2



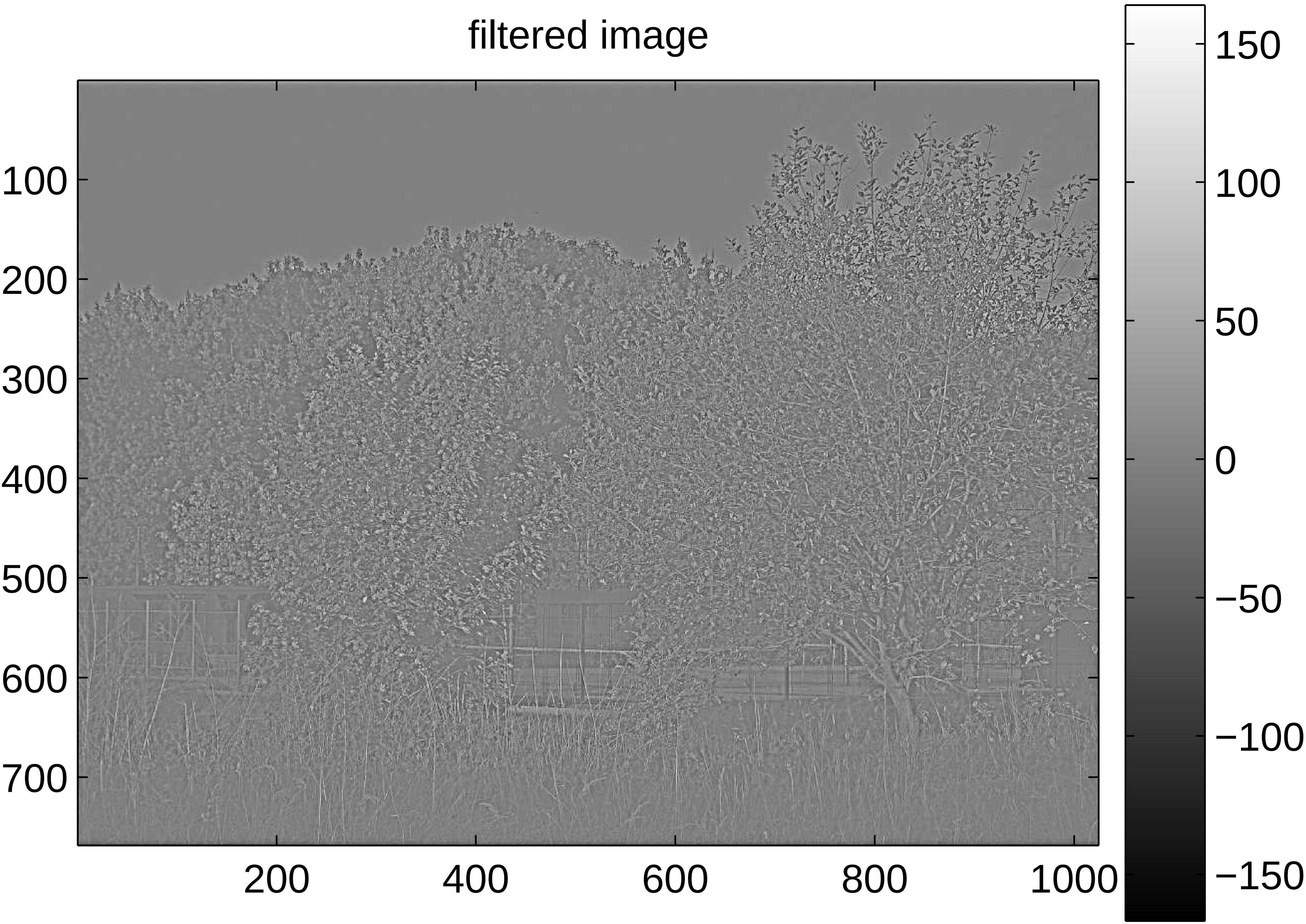
filtered image
100
200
300
400
500
600
700
200
400
600
800
1000
150
100
50
0
−50
−100
−150

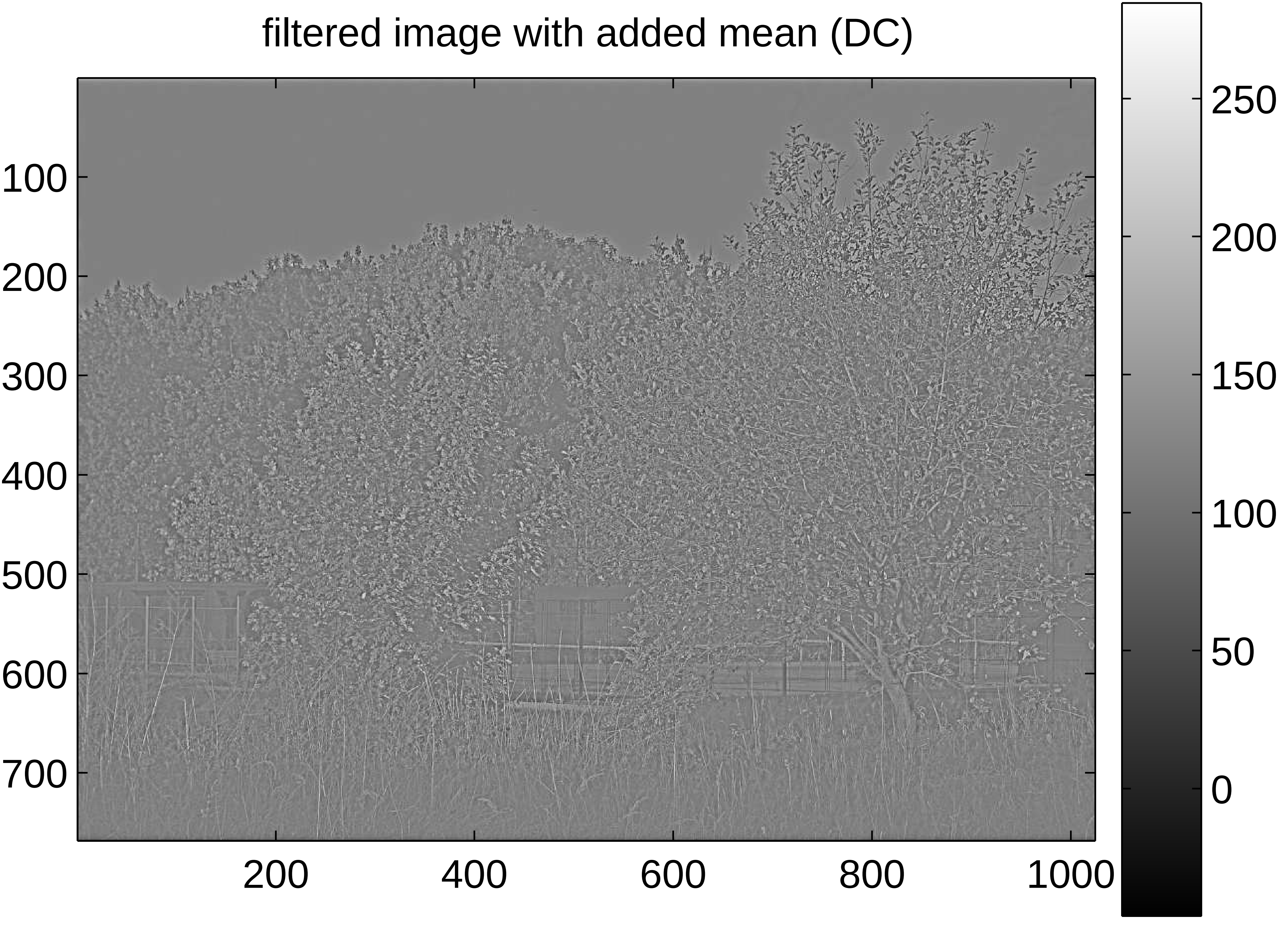

filtered image with added mean (DC)
100
200
300
400
500
600
700
200
400
600
800
1000
250
200
150
100
50
0

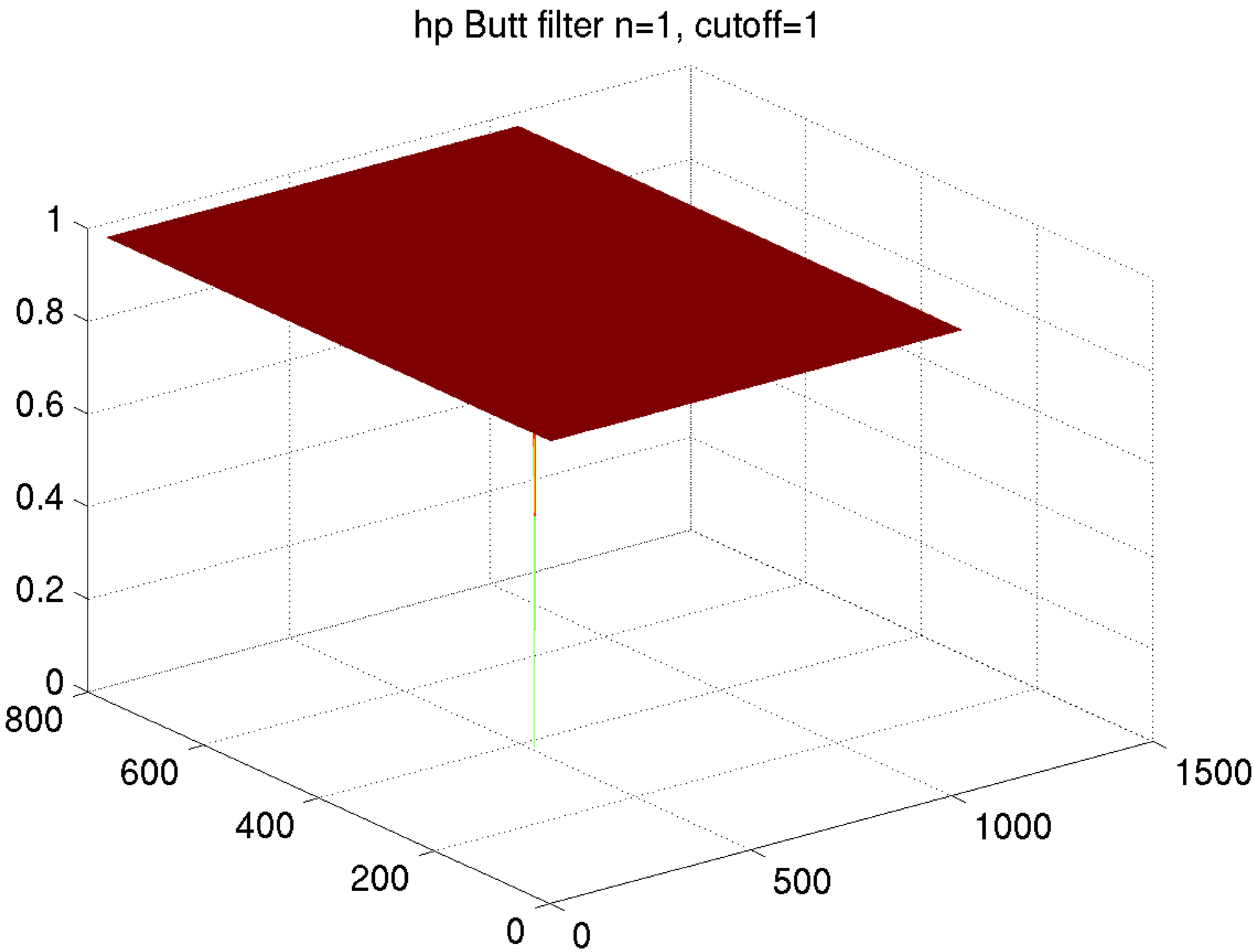

filtered image
100
200
300
400
500
600
700
200
400
600
800
1000
150
100
50
0
−50
−100
−150



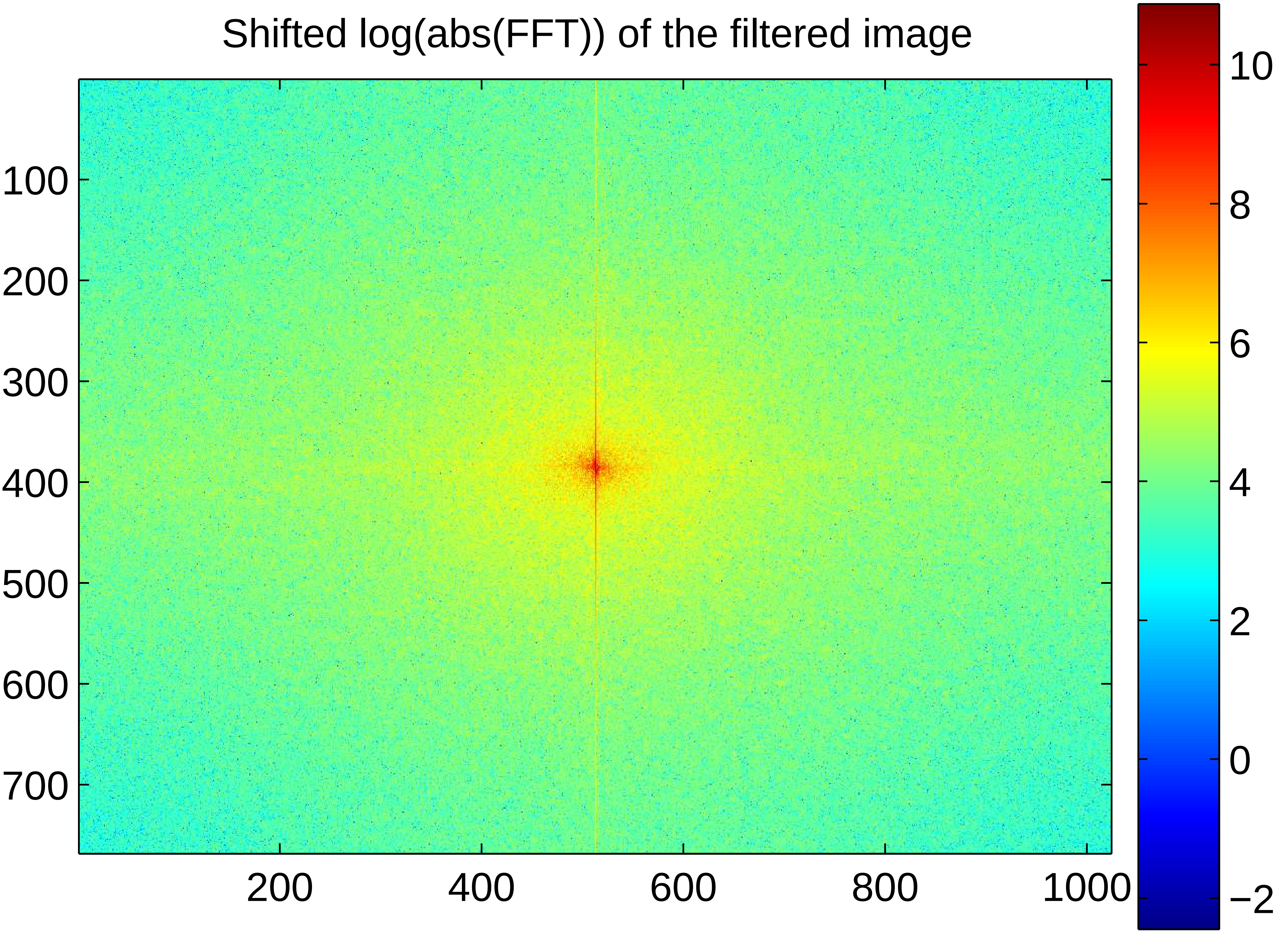
filtered image with added mean (DC)
100
200
300
400
500
600
700
200
400
600
800
1000
250
200
150
100
50
0

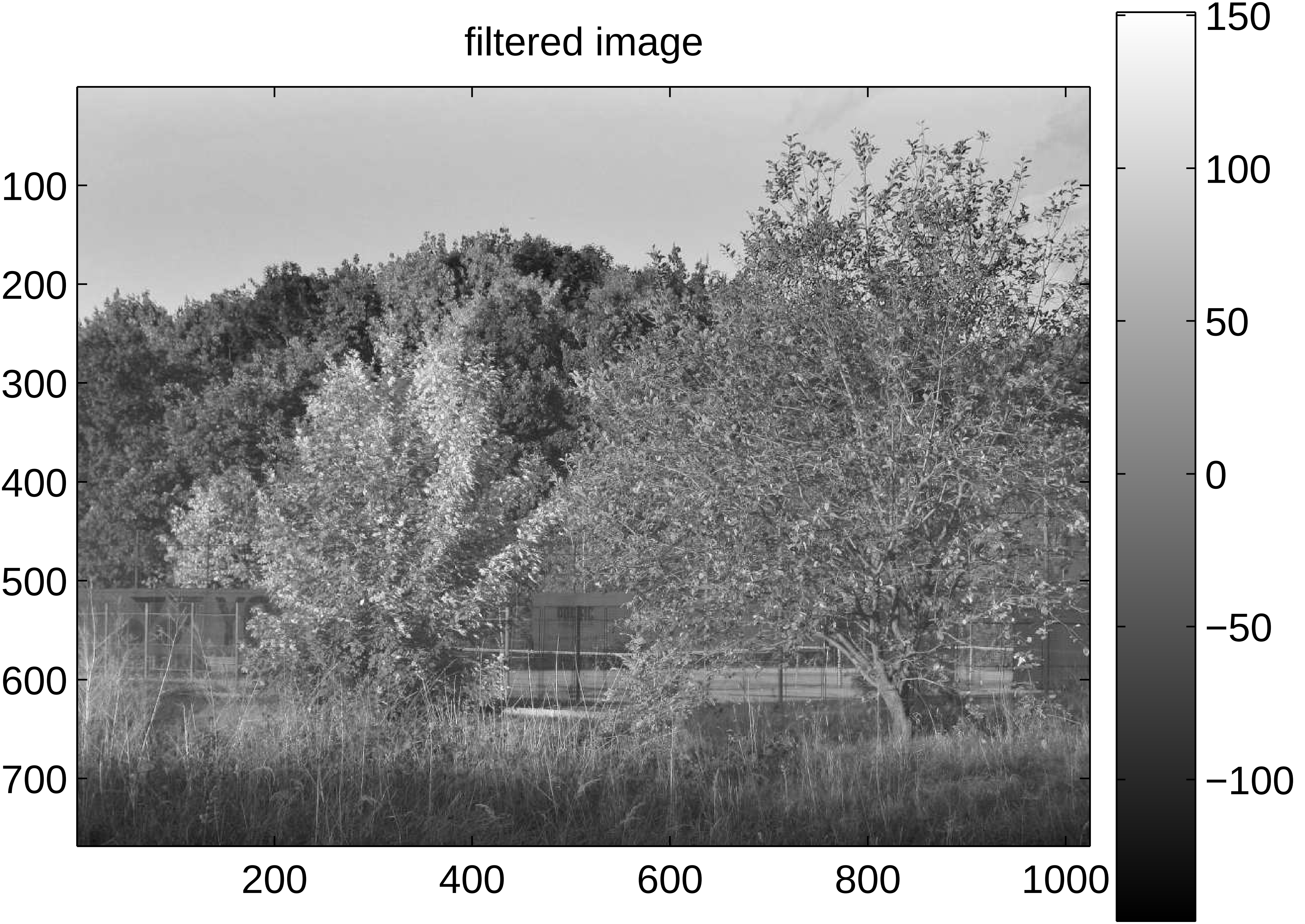

filtered image
100
200
300
400
500
600
700
200
400
600
800
1000
150
100
50
0
−50
−100
−150

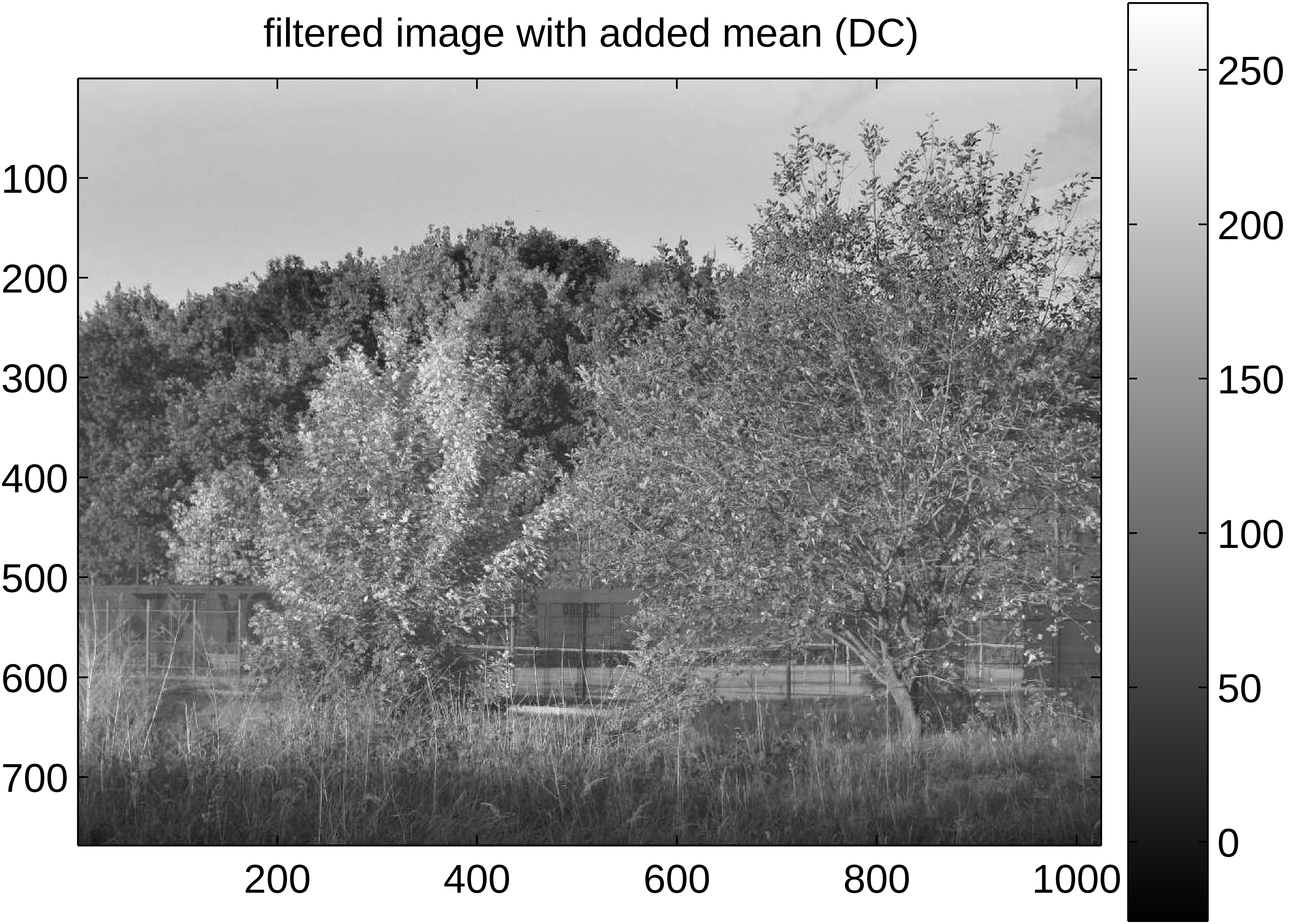

filtered image with added mean (DC)
100
200
300
400
500
600
700
200
400
600
800
1000
250
200
150
100
50
0



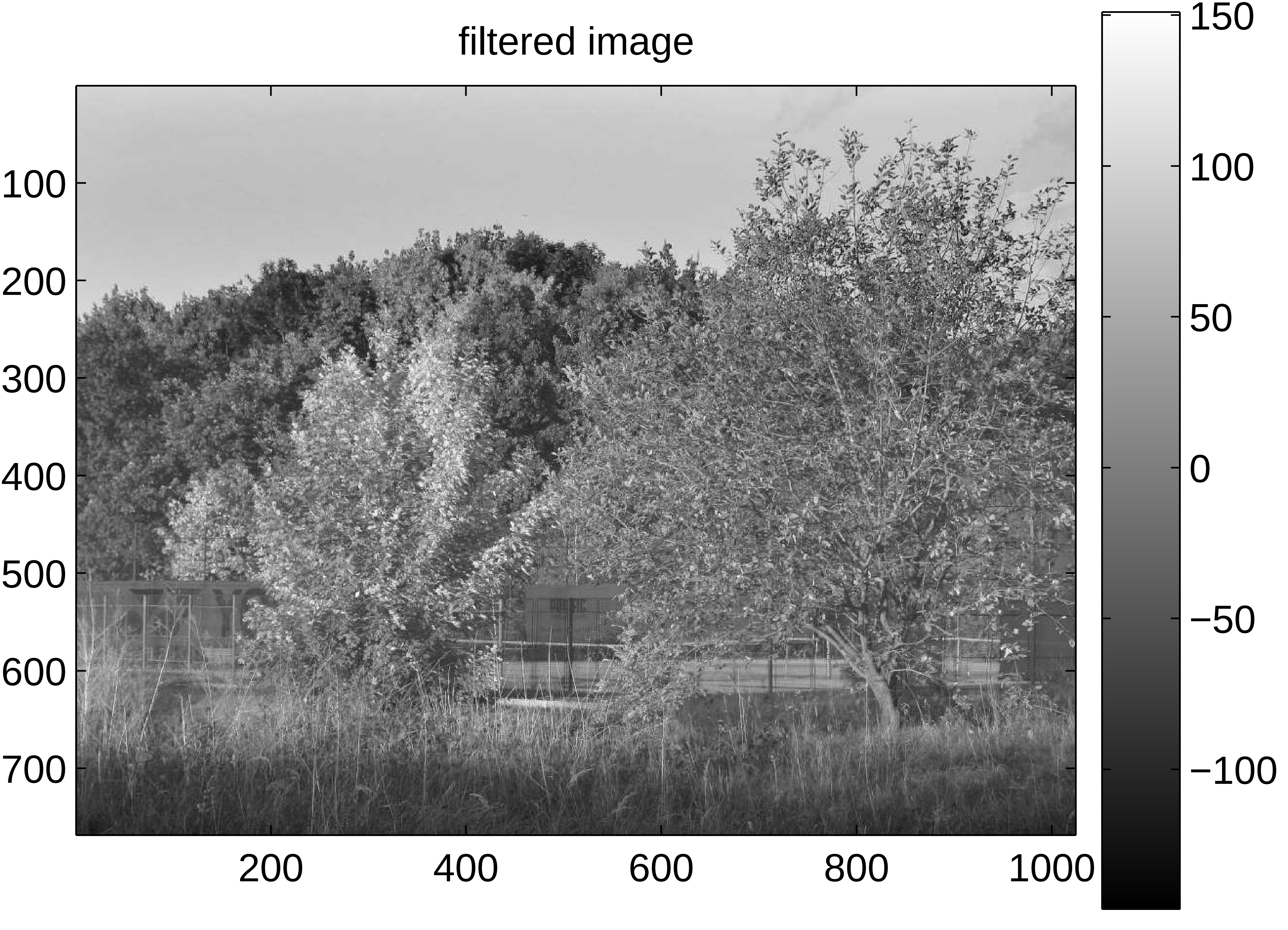
hp Butt filter n=1, cutoff=1
1
0.8
0.6
0.4
0.2
0
800
600
400
200
0
0
500
1000
1500

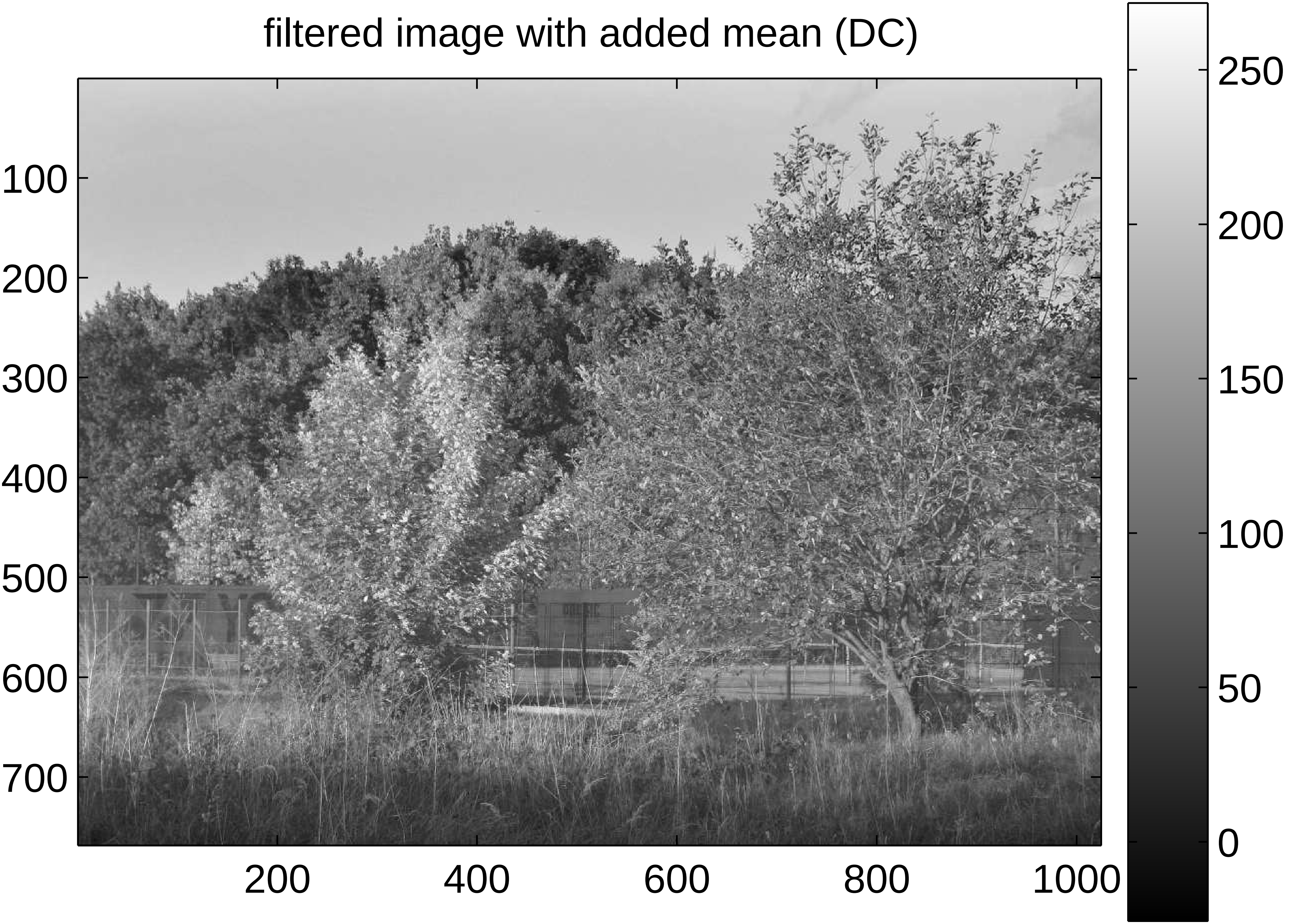

Shifted log(abs(FFT)) of the filtered image
100
200
300
400
500
600
700
200
400
600
800
1000
10
8
6
4
2
0
−2

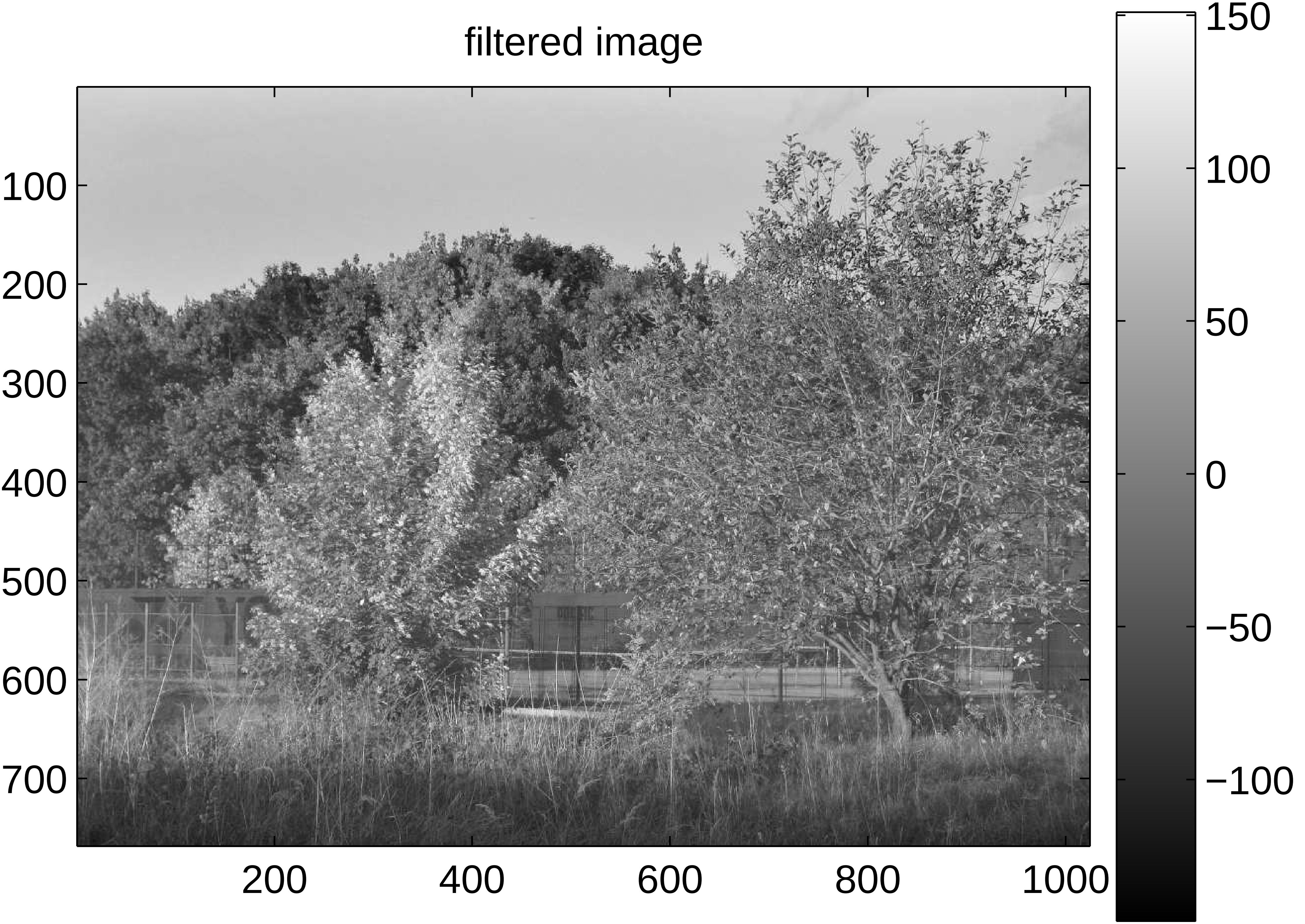

filtered image
100
200
300
400
500
600
700
200
400
600
800
1000
150
100
50
0
−50
−100

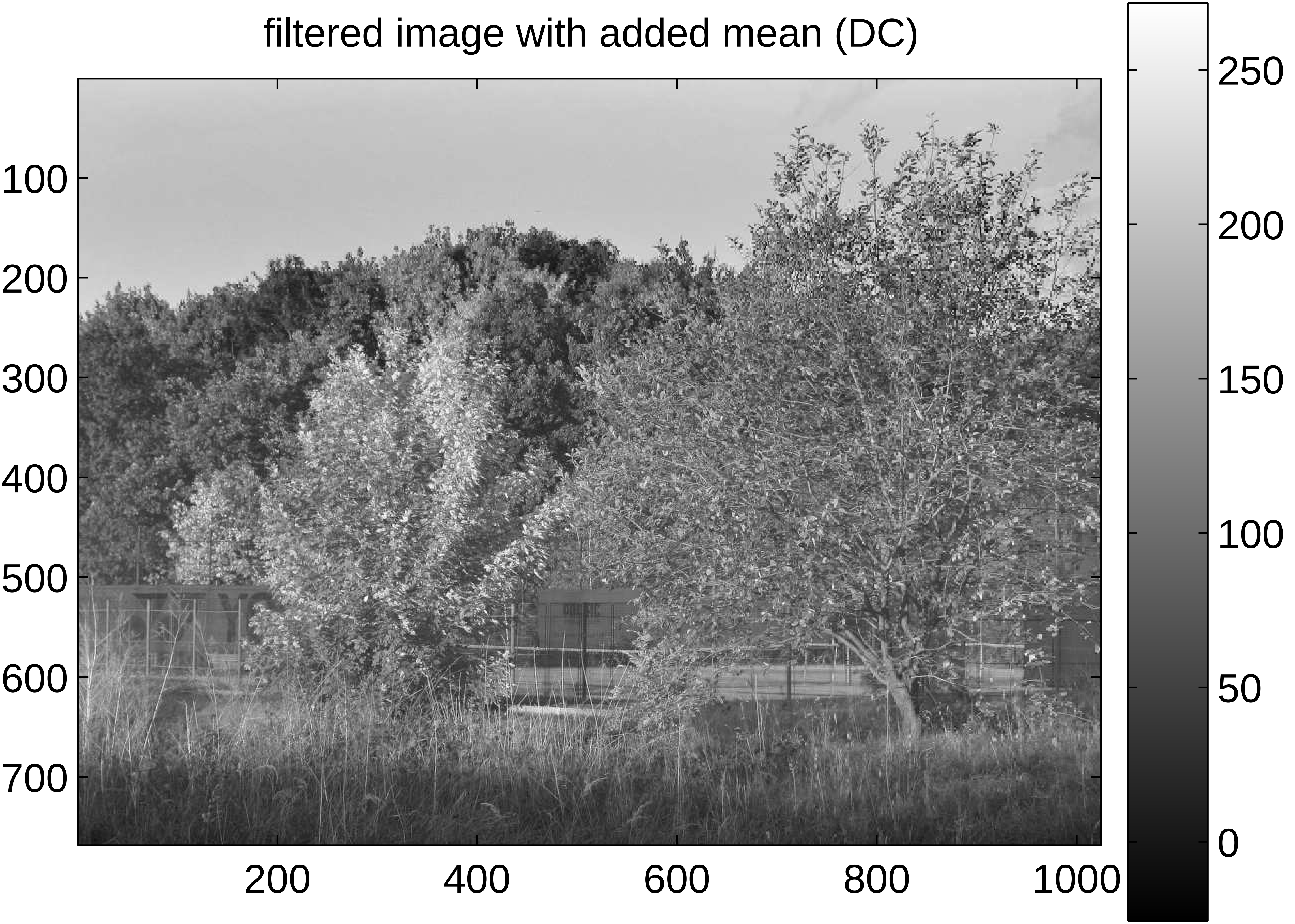


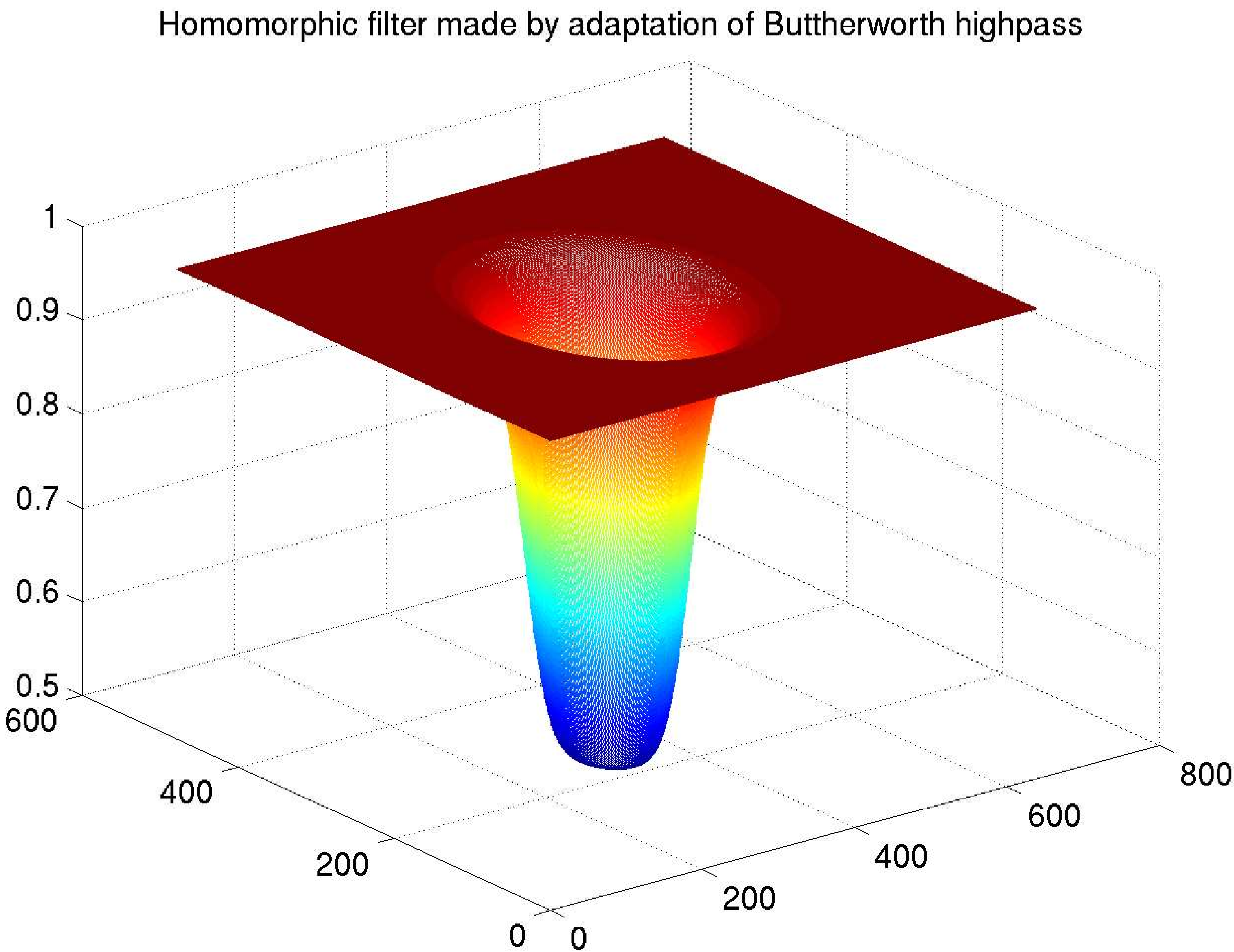
filtered image with added mean (DC)
100
200
300
400
500
600
700
200
400
600
800
1000
250
200
150
100
50
0

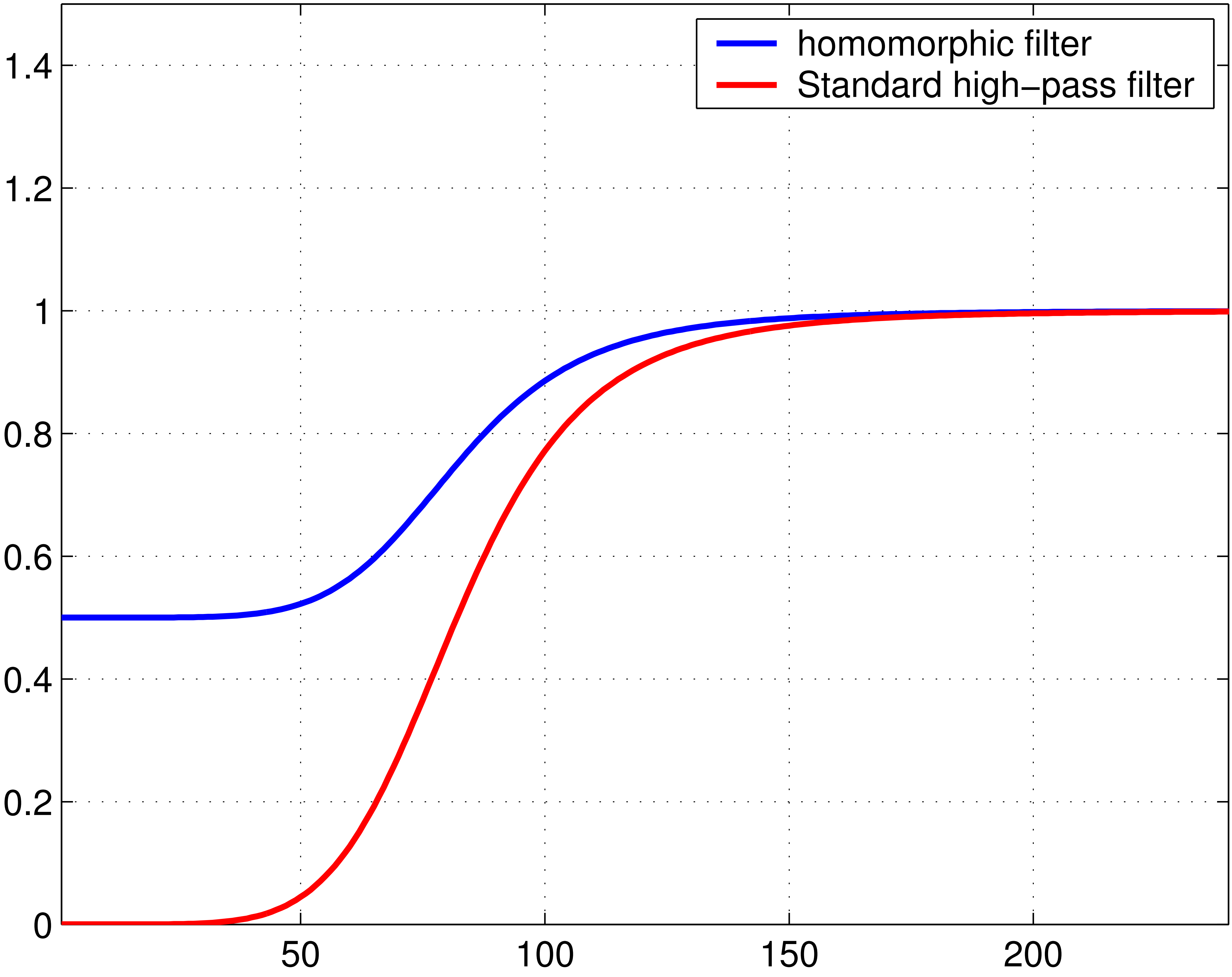

filtered image
100
200
300
400
500
600
700
200
400
600
800
1000
150
100
50
0
−50
−100


filtered image with added mean (DC)
100
200
300
400
500
600
700
200
400
600
800
1000
250
200
150
100
50
0


filtered image
100
200
300
400
500
600
700
200
400
600
800
1000
150
100
50
0
−50
−100


filtered image with added mean (DC)
100
200
300
400
500
600
700
200
400
600
800
1000
250
200
150
100
50
0


Homomorphic filter made by adaptation of Buttherworth highpass
1
0.9
0.8
0.7
0.6
0.5
600
400
200
0
0
200
400
600
800


homomorphic filter
Standard high−pass filter
1.4
1.2
1
0.8
0.6
0.4
0.2
0
50
100
150
200