In [2]:	<pre>import matplotlib.pyplot as plt import numpy as np</pre>
Tn 2 :	Densité spectrale du bruit d'une machine à laver
In [3]:	<pre>samplerate, amplitude = read('data/machine_a_laver.wav')</pre>
111 [0].	<pre>delta_t = 1/samplerate # Time between two sample N = len(amplitude) # Nombre de points print(N) T = N*delta_t #Temps total d'enregistrement</pre>
	print(T) 44100 414380
In [4]:	
0	<pre>plt.plot(t[-samplerate:], amplitude[-samplerate:])# Plot the last 1 second of recording plt.xlabel('Temps [s]') plt.ylabel('Amplitude')</pre>
Out[4]:	0.100 - 0.075
	0.050 -
	0.000 - -0.025 -
	-0.050 -0.075 -8.4 8.6 8.8 9.0 9.2 9.4
In []:	Temps [s]
In [5]:	<pre>from scipy.signal import periodogram freq, psd = periodogram(amplitude, samplerate) # Calculate the PSD (Power Spectral Density)</pre>
In [6]:	<pre>mask = (freq>=10) & (freq<50) # Frequencies between 10 and 50Hz plt.loglog(freq[mask], psd[mask]) plt.grid(which='both')</pre>
	<pre>mask_pic = np.abs(freq-20)<1.5 # Frequencies around the peak of signal at 20 Hz (20 +/- 1.5 Hz) plt.loglog(freq[mask& ~mask_pic], psd[mask& ~mask_pic]) plt.loglog(freq[mask_pic], psd[mask_pic])</pre>
Out[6]:	
	10^{-4} 10^{-5} 10^{-6}
	10-8
	10 ⁻⁹
In [7]:	print(np.var(amplitude)) # Mean power of the recording (Signal + noise) calculated on the time data
	Delta_f = 1/T print(np.sum(psd)*Delta_f) # Mean power of the recording (Signal + noise) calculated from the PSD (should be the condition of the recording (Signal + noise) calculated from the PSD (should be the condition of the recording (Signal + noise) calculated from the PSD (should be the condition of the recording (Signal + noise) calculated from the PSD (should be the condition of the recording (Signal + noise) calculated from the PSD (should be the condition of the recording (Signal + noise) calculated from the PSD (should be the condition of the recording (Signal + noise) calculated from the PSD (should be the condition of the recording (Signal + noise) calculated from the PSD (should be the condition of the recording (Signal + noise) calculated from the PSD (should be the condition of the recording (Signal + noise) calculated from the PSD (should be the condition of the
In [8]:	<pre>print('Signal', np.sum(psd[mask_pic]*Delta_f)) # Mean power of the signal print('Bruit', np.sum(psd[~mask_pic])*Delta_f) # Mean power of the noise > signal! Signal 3.569248786907483e-05</pre>
In [9]:	Bruit 0.0005723846335586126
In [12]:	0.10642405521502003 0.10642405521502003 mask = (freq>=10) & (freq<500) plt.loglog(freq[mask], psd[mask]) # Plot on a larger frequncy scale,
Out[12]:	[<matplotlib.lines.line2d 0x7f729f36cee0="" at="">]</matplotlib.lines.line2d>
	10-4 -
	10-8 -
	10-10 -
	Filtre en Python
In [14]:	samplerate = 44100 delta_t = 1/samplerate
	<pre>signal = np.zeros(samplerate*3) signal[samplerate:samplerate*2] = 1 # A temporal rectangular function t = np.arange(len(signal))*delta_t plt.plot(t, signal)</pre>
Out[14]:	[<matplotlib.lines.line2d 0x7f729f231ac0="" at="">]</matplotlib.lines.line2d>
	0.8 -
	0.4 -
	0.0 0.5 1.0 1.5 2.0 2.5 3.0
In [16]:	<pre>import numpy as np signal_tilde = np.fft.rfft(signal) # Fourier transform of the rectangular function signal_2 = np.fft.irfft(signal_tilde) # inverse Fourier transform of fourier transform = rectangular function</pre>
Out[16]:	<pre>plt.plot(t, signal_2)</pre>
	0.8
	0.6 -
	0.2 -
In [17]:	
In [17]: Out[17]:	# 1/T => samplerate/2
In [18]:	<pre>def pass_bas(signal, f_c, samplerate=44100): signal_tilde = np.fft.rfft(signal) # Fourier transform freqs = np.fft.rfftfreq(len(signal), 1/samplerate) #Frequency scale</pre>
-	<pre>H = 1/(1+1J*(freqs/f_c)) # Transfer function of the low pass filter signal_2 = np.fft.irfft(H*signal_tilde) # Filtered temporal signal return signal_2</pre>
In [19]:	<pre>def pass_haut(signal, f_c, samplerate=44100): signal_tilde = np.fft.rfft(signal) # Fourier transform freqs = np.fft.rfftfreq(len(signal), 1/samplerate) #Frequency scale H = 1J*(freqs/f_c)/(1+1J*(freqs/f_c)) # Transfer function of the high pass filter signal_2 = np.fft.irfft(H*signal_tilde) # Filtered temporal signal</pre>
In [20]:	<pre>return signal_2 signal_filtre = pass_bas(signal, f_c=3) # Low pas filtered rectangular function, cutoff at 3Hz</pre>
Out[20]:	plt.plot(t, signal_filtre) [<matplotlib.lines.line2d 0x7f729f10f400="" at="">]</matplotlib.lines.line2d>
	0.8 -
	0.6 -
	0.2
In [21]:	signal_filtre = pass_haut(signal, f_c=3) # High pas filtered rectangular function, cutoff at 3Hz
Out[21]:	
	1.00 - 0.75 - 0.50 -
	0.25 - 0.00 - -0.25 -
	-0.50 - -0.75 - -1.00 -
In [36]:	from scipy.io.wavfile import read samplerate, amplitude = read('data/machine_a_laver.wav')
To [07].	amplitude_filtree = pass_haut(pass_bas(amplitude, 30, samplerate=samplerate),15, samplerate=samplerate) #High &
In [37]:	<pre>t = np.arange(N)*delta_t plt.plot(t[-44100:], amplitude[-44100:]) plt.plot(t[-44100:], amplitude_filtree[-44100:]) # We can see the temporal washing machine signal at 20Hz after plt.xlabel('Temps [s]')</pre>
Out[37]:	plt.ylabel('Amplitude')
	0.100 - 0.075 - 0.050 -
	9 0.025 - 0.000 - 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
	-0.025 - -0.050 - -0.075 -
	8.4 8.6 8.8 9.0 9.2 9.4 Temps [s]
In [24]:	
In [25]:	
	filename_H1 = 'data/H-H1_LOSC_4_V1-1126259446-32.hdf5' strain_H1, time_H1, chan_dict_H1 = loaddata(filename_H1, 'H1') #Temporal signal from 1st detector
	filename_L1 = 'data/L-L1_LOSC_4_V1-1126259446-32.hdf5' strain_L1, time_L1, chan_dict_L1 = loaddata(filename_L1, 'L1') #Temporal signal from 2nd detector
	<pre>strain_L1, time_L1, chan_dict_L1 = loaddata(filename_L1, 'L1') #Temporal signal from 2nd detector time = time_H1 # Time print(len(time)) dt = time[1] - time[0]</pre>
	<pre>strain_L1, time_L1, chan_dict_L1 = loaddata(filename_L1, 'L1') #Temporal signal from 2nd detector time = time_H1 # Time print(len(time))</pre>
In [26]:	<pre>strain_L1, time_L1, chan_dict_L1 = loaddata(filename_L1, 'L1') #Temporal signal from 2nd detector time = time_H1 # Time print(len(time)) dt = time[1] - time[0] print(dt) samplerate = int(1/dt) 131072 0.000244140625</pre>
In [26]:	<pre>strain_L1, time_L1, chan_dict_L1 = loaddata(filename_L1, 'L1') #Temporal signal from 2nd detector time = time_H1 # Time print(len(time)) dt = time[1] - time[0] print(dt) samplerate = int(1/dt) 131072 0.000244140625 # plot +- 5 seconds around the event: tevent = 1126259462 # Mon Sep 14 09:50:45 GMT 2015</pre>
	<pre>strain_L1, time_L1, chan_dict_L1 = loaddata(filename_L1, 'L1') #Temporal signal from 2nd detector time = time_H1 # Time print(len(time)) dt = time[1] - time[0] print(dt) samplerate = int(1/dt) 131072 0.000244140625 # plot +- 5 seconds around the event: tevent = 1126259462 # Mon Sep 14 09:50:45 GMT 2015 deltat = 5 # seconds around the event # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) plt.plot(time_H1[mask] - tevent, strain_H1[mask]) #We see that we have parasite oscillations and noise, we can</pre>
In [26]:	<pre>strain_L1, time_L1, chan_dict_L1 = loaddata(filename_L1, 'L1') #Temporal signal from 2nd detector time = time_H1 # Time print(len(time)) dt = time[1] - time[0] print(dt) samplerate = int(1/dt) 131072 0.000244140625 # plot +- 5 seconds around the event: tevent = 1126259462 # Mon Sep 14 09:50:45 GMT 2015 deltat = 5 # seconds around the event # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) plt.plot(time_H1[mask] - tevent, strain_H1[mask]) #We see that we have parasite oscillations and noise, we can [<matplotlib.lines.line2d 0x7f729e8fe520="" at="">]</matplotlib.lines.line2d></pre>
	<pre>strain_L1, time_L1, chan_dict_L1 = loaddata(filename_L1, 'L1') #Temporal signal from 2nd detector time = time_H1 # Time print(len(time)) dt = time[1] - time[0] print(dt) samplerate = int(1/dt) 131072 0.000244140625 # plot +- 5 seconds around the event: tevent = 1126259462 # Mon Sep 14 09:50:45 GMT 2015 deltat = 5 # seconds around the event # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) plt.plot(time_H1[mask] - tevent, strain_H1[mask]) #We see that we have parasite oscillations and noise, we car [<matplotlib.lines.line2d 0x7f729e8fe520="" at="">] le-19</matplotlib.lines.line2d></pre>
	<pre>strain_L1, time_L1, chan_dict_L1 = loaddata(filename_L1, 'L1') #Temporal signal from 2nd detector time = time_H1 # Time print(len(time)) dt = time[1] - time[0] print(dt) samplerate = int(1/dt) 131072 0.000244140625 # plot +- 5 seconds around the event: tevent = 1126259462 # Mon Sep 14 09:50:45 GMT 2015 deltat = 5 # seconds around the event # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) plt.plot(time_H1[mask] - tevent, strain_H1[mask]) #We see that we have parasite oscillations and noise, we can [<matplotlib.lines.line2d 0x7f729e8fe520="" at="">]</matplotlib.lines.line2d></pre>
	<pre>strain_L1, time_L1, chan_dict_L1 = loaddata(filename_L1, 'L1') #Temporal signal from 2nd detector time = time_H1 # Time print(len(time)) dt = time[1] - time[0] print(dt) samplerate = int(1/dt) 131072 0.000244140625 # plot +- 5 seconds around the event: tevent = 1126259462 # Mon Sep 14 09:50:45 GMT 2015 deltat = 5 # seconds around the event # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) plt.plot(time_H1[mask] - tevent, strain_H1[mask]) #We see that we have parasite oscillations and noise, we can [<matplotlib.lines.line2d 0x7f729e8fe520="" at="">]</matplotlib.lines.line2d></pre>
	strain_L1, time_L1, chan_dict_L1 = loaddata(filename_L1, 'L1') #Temporal signal from 2nd detector time = time_H1 # Time print(len(time)) dt = time[1] - time[0] print(dt) samplerate = int(1/dt) 131072 0.000244140625 # plot +- 5 seconds around the event: tevent = 1126259462 # Mon Sep 14 09:50:45 GMT 2015 deltat = 5 # seconds around the event # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) plt.plot(time_H1[mask] - tevent, strain_H1[mask]) #We see that we have parasite oscillations and noise, we can [<matplotlib.lines.line2d 0x7f729e8fes20="" at="">] 1e-19 1</matplotlib.lines.line2d>
Out[26]:	strain_Li, time_Li, chan_dict_Li = loaddata(filename_Li, 'Li') #Tomporal signal from 2nd detector time = time_Hi * Time print(len(time)) dt = time[i] * time[0] print(dt) samplerate = int(1/dt) 131072 0.080244140625 # plot +- 5 seconds around the event: tevent = 1126259462 # Mon Sep 14 09:59:45 GMT 2015 deltat = 5
Out[26]:	strain_L1, time_L1, chan_dict_L1 = loaddata(filename_L1, 'L1') #Temporal signal from 2nd detector time = time_H1 # Time print(len(time)) dt = time[i] * time[0] print(di) samplerate = int(1/dt) 131072 0.000244140625 # plot +- 5 seconds around the event: tevent = 1120259402 # Mon Sep 14 09:50:45 GMT 2015 deltat = 5 # seconds around the event # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) # index into the strain time series for this time interval: mask = ((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat)) plt.plot(time_H1[mask] - tevent, strain_H1[mask]) #We see that we have parasite oscillations and noise, we can [<matplotlib.lines.line2d 0x7f729e8fe520="" at="">] **Prom scipy.signal import periodogram f_p, psd_p = periodogram(strain_H1, samplerate) plt.loglog(f_p[300:], psd_p[300:]) # PSD of the measured signal, we see noise and parasite signals as sharp pea</matplotlib.lines.line2d>
Out[26]:	strain_Li, time_Li, chan_dict_Li = loaddata(filename_Li, 'Li') #Temporal signal from 2nd detector time = time_Hi # Time print(len(time)) dt = time[i] - time[e] print(di) samplerate = int(1/dt) 131877 0.080244140625 # plot + - S seconds around the event: tovent = 112625962 # **Mon Sop 14 09:59:45 GMT 2015 deltat = S
Out[26]:	strain_Li, time_Li, chan_dict_Li = loaddata(filename_Li, 'Li') #Temporal signal from 2nd detector time = time_Hi # Time print(len(time)) dt = time[i] * time[0] print(cit) samplerate = int(1/dt) 13i872 8.080244140625 # plot +- 5 seconds around the event: tevent = 11626250402 # Mon Sep 14 09:58:45 GNT 2015 deltat = 5 # seconds around the event # index into the strain time series for this time interval: mask = ((time_Hi >= tevent-deltat) & ((time_Hi < tevent-deltat)) # index into the strain time series for this time interval: mask = ((time_Hi >= tevent-deltat) & (time_Hi < tevent-deltat)) plt.plot(time_Hi[mask] - tevent, strain_Hi[mask]) #We see that we have parasite oscillations and noise, we can [<matplotlib.lines.line2d 0x7f729e8fe520="" at="">] # loglog(f_p[500:], psd_p[500:]) # PSD of the measured signal, we see noise and parasite signals as sharp pea [<matplotlib.lines.line2d 0x7f729e8da040="" at="">] 10-14 1</matplotlib.lines.line2d></matplotlib.lines.line2d>
Out[26]:	strain_11, time_11, chan_dict_11 = loaddata(filename_11, 'l1') #Temporal signal from 2nd detector time = time H1 # Time print(len(time)) dt = time(1) - time(0) dt = time(1) - time(1) dt = time(1) - time(0) dt = time(1) - time(1) dt = time
Out[26]: In [27]: In [28]:	strain Li, time Li, chan dict.Li = losddata(filename Li, 'Li') #Temporal signal from 2nd detector time = time Hi * 7 fme print(Lon(Line)) print(dit)
Out[26]: In [27]: Out[27]:	strain L1, time L1, chan dict L1 = loaddata(filename L1, 'L1') #Temporal signal from 2nd detector print(len(time)) print(len(
Out[26]: In [27]: In [28]:	strain_tl, time_tln_cham_dist_tl = loaddata(filmnam_tl, 'tl') #Jomporal signal from 2nd detector time_time_time_time.) dr = time_tln(time_time) dr = time_tln(time_time) print_(dt) zamplerate = int(1/dt) 134672 # plot : 5 seconds around the event: detect = 5 seconds around the event: detect = 5 seconds around the event # index sind the strain time sories for this (time interval): mask = ((time_tln = cevent_doll(at) & (time_tln < tevent_doll(at))) # index sind the strain time secies for this time interval: mask = ((time_tln > tevent_doll(at)) & (time_tln < tevent_doll(at))) # index sind the strain time secies for this time interval: mask = (time_tln > tevent_doll(at)) & (time_tln < tevent_doll(at))) # index sind the strain time secies for this time interval: mask = (time_tln > tevent_doll(at)) & (time_tln < tevent_doll(at))) # index sind the strain time secies for this time interval: mask = (time_tln > tevent_doll(at)) & (time_tln < tevent_doll(at))) # index sind the strain time secies for this time interval: mask = (time_tln > time_tln < time_t
Out[26]: In [27]: In [28]:	strain_tis_time_tis_time_tis_ time = tism_N = 7.5m prant[cln(time)) dr = time[1] = time[d] print(dr) 2.00034410620 # plot + 5 seconds around the event: twonn = 120220402 # Non Sen 10 00:50:45 GHT 2015 deltat = time[1] = time[d] # non Sen 10 00:50:45 GHT 2015 # index into the strain time series for this (time interval: mask = {(time_H) = (cond_obleta) & (time_H) < (twent_obleta)) # index into the strain time series for this (time interval: mask = {(time_H) = (cond_obleta) & (time_H) < (twent_obleta)) # index into the strain time series for this (time interval: mask = {(time_H) = (cond_obleta) & (time_H) < (twent_obleta)) # index into the strain time series for this (time interval: mask = {(time_H) = (cond_obleta) & (time_H) < (tevent_obleta)) # index into the strain time series for this (time interval: mask = {(time_H) = (cond_obleta) & (time_H) < (time_time_val: mask = {(time_H) = (cond_obleta) & (time_H) < (time_time_val: mask = {(time_H) = (cond_obleta) & (time_H) < (time_time_val: # pod_p = periodogram(strain_H), samplerate) plt.loglof(_sign_H) = pod_p(sen_H) = Fen_H obleta # pod_p = periodogram(strain_H), samplerate) # prod_p = periodogram(strain_H), samplerate, prof_p = periodogram(strain_H), samplerate) # prof_p = periodogram(strain_H), samplerate, prof_p = periodogram(strain_H), samplerate) # prof_p = periodogram(strain_H), samplerate, prof_p = periodogram(strain_H), samplerate) # prof_p = periodogram(strain_H), samplerate, prof_p = periodogram(strain_H), samplerate) # prof_p = periodogram(strain_H), samplerate, prof_p = periodogram(strain_H), samplerate) # prof_p = periodogram(strain_H), samplerate, prof_
Out[26]: In [27]: In [28]:	Strain 13, time 13, plan dist it = loaddata(filename i1, 'i1') #imagora? signal from 2nd delector time = time, Mark & Time print(low(time)) dt = time[1] print(dt) samplerate = int(fult) 131872 # plot + 6 seconds around the ovent: tevent = 118259462 # Non Sep 14 89:50:45 SMT 2015 deltat = 5 # seconds around the event mask = (filen_Hi >= tevent-deltat) & (time_Hi <= tevent+deltat)) # shows afton the strain time series for that time interval: mask = (filen_Hi >= tevent-deltat) & (time_Hi <= tevent+deltat)) # shows afton the strain time series for that time sine series mask = (filen_Hi >= tevent-deltat) & (time_Hi <= tevent+deltat)) # plic.plot(time_Hi mask) - tevent, strain_Hi[mask]) **whe see that we have parasite oscillations and noise, we call constplot(iis.lines.line2D at 8x77720eds8640>) # plic.plot(iis.lines.line2D at 8x77720eds8640>) # promote the second size of the second size of the measured size of the second parasite size of averaging but we let # plic.plot(iis.lines.line2D at 8x77720eds8640>) # promote the second size of the second size of averaging but we let # plic.plot(iis.lines.line2D at 8x7720edf8360>)
Out[26]: In [27]: In [28]:	Strants, time_ts, etc., then_dict_ts = loaddste(filename_ts, "ts") #Temporal aignal from 2nd detector time = time_ts = Time Time_ts print(len(time)) distant_1 = Time_ts print(len(time)) distant_1 = Time_ts print(len(time)) distant_2 = Time_ts print(len(time)) distant_3 = Time_ts distant_3 = Time
Out[26]: In [27]: Out[27]:	strain_li, Line_Li, chan_dict_Li = lossata(flienene_Li, 'li') Flemonal signal from 2nd detector time = Line M is fine print(Den(time)) of the Cine[1] - Line[0] print(did Line[0] print(
Out[26]: In [27]: Out[27]:	itrac_li, time_li, employed, a Time state_li, employed, and the continue to th
Out[26]: In [27]: Out[27]:	print [1], then [1], case direct [1] = loadstate(*Ilenamo [1], "19") a Tangara? Algosi from Pool distance time = time[1] = time[8] symplews = lin(2)(4) 2002 0.00024200028 # Second around the exect: Level = 120000802
Out[26]: In [27]: Out[27]:	principal final lines. Linego at soft Time important principal final final detector that is required to the second around the sweet;
Out[26]: In [27]: Out[27]:	stront, i., camp. I., chem. principle = isoaddras(tilonomo.i.i., "Li") = Tomorani signal tree Zon decorar time = Liming. Finding.
Out[26]: In [27]: Out[27]:	remain 1. time, Li, chem, fair-Li = landersof(timeranc_Li, "Li") Stemperal signal from 2nd detector rems = Thom, facilities price(incline)
Out[26]: In [27]: Out[27]:	Street, Live, Li, checked, Li = leadate(filment, Li, "Li") of proposal signal from 2nd detector the = Targe, Live (1) Live(2) Supplied = Live(2) Supplied = Live(3) Live(2) Live(3) Live(3
Out[26]: In [27]: Out[28]:	Stant L. Live L., June dis. I. = Improbatifilmore L., "L1") "Processis signed from the detector print (red (lev)) "responding liver (lev)" processing the print (red (lev)) "responding liver (lev)" processing and the answer (lev) process of the control of the co
Out[26]: In [27]: Out[27]:	Stant L. Live L., June dis. I. = Improbatifilmore L., "L1") "Processis signed from the detector print (red (lev)) "responding liver (lev)" processing the print (red (lev)) "responding liver (lev)" processing and the answer (lev) process of the control of the co
Out[26]: In [27]: Out[28]:	Sition to C. the D. the first of the processing of the control of the processing of the processing of the control of the contr
Out[26]: In [27]: Out[28]:	Screen, J. (1961), planty all planty (1971) and the process of the
Out[26]: In [27]: Out[28]:	Stratural, Timush, descripting to account of the manual, the control about focus and description of the control account of the control ac
Out[26]: In [27]: Out[27]: In [29]:	Translate, freed, a beneficial and continued
Out [26]: In [27]: Out [28]: In [29]:	The color facts isseed specially income and controlled to the color product of the color prod
Out [26]: In [27]: Out [28]: In [29]:	The State of the S
Out [26]: In [27]: Out [28]: In [29]:	Second Lat. Care Lat. Once 18 Jacobs 20 Jacobs
Out [26]: In [27]: Out [28]: In [29]:	STATE A. TILLE II. COM SEC. 1 - Books 1 (1997) Processed signal from the discretion of the company of the compa
Out [26]: In [27]: Out [28]: In [29]:	Straff LL Car LL con data La Bandhad (Harmon LL) Alexandral Speak from the description of the control of the c
Out [26]: In [27]: Out [28]: In [29]:	Section 1, 15th 1, 15t
Out [26]: In [27]: Out [28]: In [29]:	The control of the co
Out[26]: In [27]: Out[27]: In [29]: In [30]:	Figure 1. The control of the control
Out[26]: In [27]: Out[27]: In [30]: In [31]: In [32]:	The control of the co
Out[26]: In [27]: Out[27]: In [30]: In [31]: In [32]:	and the content of th
Out[26]: In [27]: Out[27]: In [30]: In [31]: In [32]:	restrict, the processing, it is understall assembly, the processing from the absence of a process of a proces
<pre>In [27]: In [27]: In [28]: In [30]: In [32]: In [34]: </pre>	district. In worth congress of a constraint (through), if the Power's supply the American Services and America
Out[26]: In [27]: Out[27]: In [30]: In [31]: In [32]:	district. In worth congress of a constraint (through), if the Power's supply the American Services and America
<pre>In [27]: In [27]: In [28]: In [30]: In [32]: In [34]: </pre>	altered. In the John Street, and a control of the c
<pre>In [27]: In [27]: In [28]: In [30]: In [32]: In [34]: </pre>	STATES, I. See Control of the Contro
<pre>In [27]: In [27]: In [28]: In [30]: In [32]: In [34]: </pre>	STATES, Land Company of the Company
<pre>In [27]: In [27]: In [28]: In [30]: In [32]: In [34]: </pre>	Learner to the control of the contro
<pre>In [27]: In [28]: In [29]: In [30]: In [34]: Out [34]: </pre>	School Land Control Land Contro
<pre>In [27]: In [28]: In [29]: In [30]: In [34]: Out [34]: </pre>	Some 14. In the other care and absolute to the
<pre>In [27]: In [28]: In [29]: In [30]: In [34]: Out [34]: </pre>	And the control of th
In [27]: In [28]: In [30]: In [34]: In [34]:	Advanced to the first of the control
In [27]: In [28]: In [30]: In [34]: In [34]:	ACCOUNT OF THE MAN AND AND AND AND AND AND AND AND AND A
In [27]: In [28]: In [30]: In [34]: In [34]:	ACCOUNTS OF THE PROPERTY OF THE PROPERTY CONTROL OF THE PROPERTY OF THE PROPER