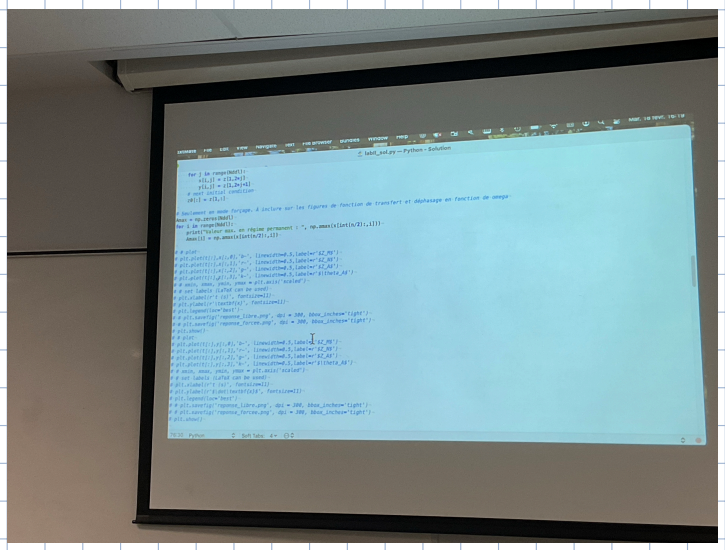
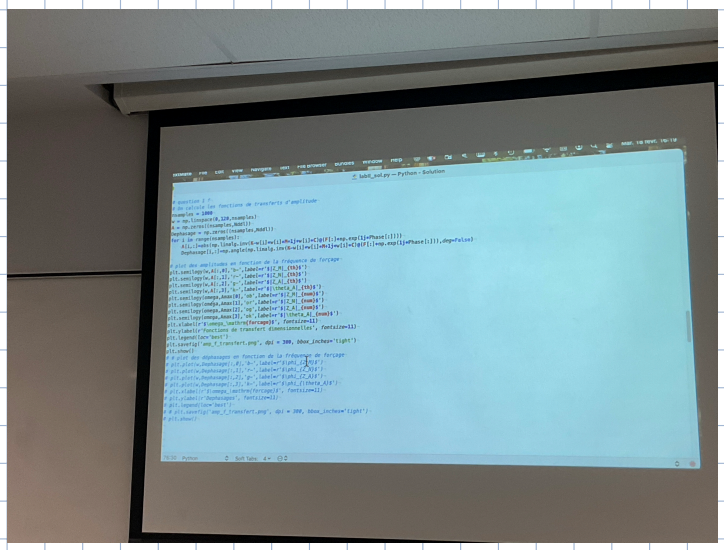

label_sol.py — Python - Solution
for i in range(Nbdt):
x[i,1] = z[i,2*j]
y[i,1] = z[i,2*j+1]
cond initial condition
z0[:] = z[1,:]
# Seulement en mode forcage. A inclure sur les figures de fonction de transfert et déphasage en fonction de omega
Amax = np.zeros(Nbdt)
for i in range(Nbdt):
print("Valeur max, en régime permanent : ", np.amax(x[int(n/2):,i]))
Amax[i] = np.amax(x[int(n/2):,i])
# # plot
# plt.plot(t[:],x[:,0],'b-', linewidth=0.5,label='Z_M')
# plt.plot(t[:],x[:,1],'r-', linewidth=0.5,label='Z_N')
# plt.plot(t[:],x[:,2],'g-', linewidth=0.5,label='Z_A')
# plt.plot(t[:],z[:,3],'k-', linewidth=0.5,label='θ_A')
# # xmin, xmax, ymin, ymax = plt.axis('scaled')
# # set labels (LaTeX can be used)
# plt.xlabel('t', fontsize=11)
# plt.ylabel('$z(t)$', fontsize=11)
# plt.legend(loc='best')
# # plt.savefig('reponse_libre.png', dpi = 300, bbox_inches='tight')
# # plt.savefig('reponse_force.png', dpi = 300, bbox_inches='tight')
# plt.show()
# # plot
# plt.plot(t[:],y[:,0],'b-', linewidth=0.5,label='Z_M')
# plt.plot(t[:],y[:,1],'r-', linewidth=0.5,label='Z_N')
# plt.plot(t[:],y[:,2],'g-', linewidth=0.5,label='Z_A')
# plt.plot(t[:],y[:,3],'k-', linewidth=0.5,label='θ_A')
# # xmin, xmax, ymin, ymax = plt.axis('scaled')
# # set labels (LaTeX can be used)
# plt.xlabel('t', fontsize=11)
# plt.ylabel('$z_{ext}(t)$', fontsize=11)
# plt.legend(loc='best')
# # plt.savefig('reponse_libre.png', dpi = 300, bbox_inches='tight')
# # plt.savefig('reponse_force.png', dpi = 300, bbox_inches='tight')
# plt.show()
78:30 Python Soft Tabs: 4


label_sol.py — Python - Solution
# question 1 :
# On calcule les fonctions de transferts d'amplitude
nsamples = 1000
w = np.linspace(0,120,nsamples)
A = np.zeros((nsamples,Nbdt))
Dephasage = np.zeros((nsamples,
for i in range(nsamples):
A[i,:]=abs(np.linalg.inv(G+w[i]*H+(w[i]*H+w[i]*C)@(F[:]*exp.exp(1j*Phase[:])))
Dephasage[i,:]=np.angle(np.linalg.inv(G+w[i]*H+w[i]*C)@(F[:]*exp.exp(1j*Phase[:])),deg=False)
# plot des amplitudes en fonction de la fréquence de forçage
plt.semilogy(w,A[:,0],'b-',label='$|Z_M|_{1%}$')
plt.semilogy(w,A[:,1],'r-',label='$|Z_N|_{1%}$')
plt.semilogy(w,A[:,2],'g-',label='$|Z_A|_{1%}$')
plt.semilogy(w,A[:,3],'k-',label='$|\theta_A|_{1%}$')
plt.semilogy(omega_Amax[0],'ob',label='$|Z_M|_{max}$')
plt.semilogy(omega_Amax[1],'or',label='$|Z_N|_{max}$')
plt.semilogy(omega_Amax[2],'og',label='$|Z_A|_{max}$')
plt.semilogy(omega_Amax[3],'ok',label='$|\theta_A|_{max}$')
plt.xlabel('$\omega_{mathrm{forcage}}$', fontsize=11)
plt.ylabel('fonctions de transfert dimensionnelles', fontsize=11)
plt.legend(loc='best')
plt.savefig('amp_f_transfert.png', dpi = 300, bbox_inches='tight')
plt.show()
# # plot des déphasages en fonction de la fréquence de forçage
# plt.plot(w,Dephasage[:,0],'b-',label='$|phi_{Z_M}|$')
# plt.plot(w,Dephasage[:,1],'r-',label='$|phi_{Z_N}|$')
# plt.plot(w,Dephasage[:,2],'g-',label='$|phi_{Z_A}|$')
# plt.plot(w,Dephasage[:,3],'k-',label='$|phi_{\theta_A}|$')
# plt.xlabel('$\omega_{mathrm{forcage}}$', fontsize=11)
# plt.ylabel('Dephasages', fontsize=11)
# plt.legend(loc='best')
# # plt.savefig('amp_f_transfert.png', dpi = 300, bbox_inches='tight')
# plt.show()
78:30 Python Soft Tabs: 4