**The lattice definition I am using is the following:**

<GRAPH name="double dimer" vertices="4">

<VERTEX id="1" type="0"></VERTEX>

<VERTEX id="2" type="0"></VERTEX>

<VERTEX id="3" type="1"></VERTEX>

<VERTEX id="4" type="1"></VERTEX>

<EDGE type="0" source="1" target="2"/>

<EDGE type="0" source="3" target="4"/>

<EDGE type="1" source="1" target="3"/>

<EDGE type="1" source="1" target="4"/>

<EDGE type="1" source="2" target="3"/>

<EDGE type="1" source="2" target="4"/>

</GRAPH>

**The Hamiltonian definition is the following:**

<SITEBASIS name="spin">

<PARAMETER name="local\_spin" default="local\_S"/>

<PARAMETER name="local\_S" default="2"/>

<QUANTUMNUMBER name="S" min="local\_spin" max="local\_spin"/>

<QUANTUMNUMBER name="Sz" min="-S" max="S"/>

<OPERATOR name="Splus" matrixelement="sqrt(S\*(S+1)-Sz\*(Sz+1))">

<CHANGE quantumnumber="Sz" change="1"/>

</OPERATOR>

<OPERATOR name="Sminus" matrixelement="sqrt(S\*(S+1)-Sz\*(Sz-1))">

<CHANGE quantumnumber="Sz" change="-1"/>

</OPERATOR>

<OPERATOR name="Sz" matrixelement="Sz"/>

</SITEBASIS>

<HAMILTONIAN name="dimerized spin">

<PARAMETER name="J" default="1"/>

<PARAMETER name="h" default="0.1"/>

<BASIS ref="spin"/>

<SITETERM site="i">

<PARAMETER name="h#" default="h"/>

-h#\*Sz(i)

</SITETERM>

<BONDTERM source="i" target="j">

<PARAMETER name="J#" default="J"/>

<PARAMETER name="h#" default="h"/>

J#\*Sz(i)\*Sz(j)+J#/2\*(Splus(i)\*Sminus(j)+Sminus(i)\*Splus(j))

</BONDTERM>

</HAMILTONIAN>

**The python script I used is the following:**

import pyalps

import matplotlib

matplotlib.use('Agg')

import matplotlib.pyplot as plt

import pyalps.plot

parms = []

parms.append(

{

'LATTICE' : "double dimer",

'MODEL' : "dimerized spin",

'LATTICE\_LIBRARY': "/home/users/tandons/alps\_test/mylat.xml",

'MODEL\_LIBRARY' : "/home/users/tandons/alps\_test/mymod.xml",

'local\_S0' : 1.0,

'local\_S1' : 0.5,

'J0' : 2,

'J1' : 0.8,

'h' : 0.202 ,

'CONSERVED\_QUANTUMNUMBERS' : "Sz",

}

)

input\_file = pyalps.writeInputFiles('parm2a',parms)

res=pyalps.runApplication('fulldiag',input\_file)

data = pyalps.evaluateFulldiagVersusT(pyalps.getResultFiles(prefix='parm2a'),DELTA\_T=10, T\_MIN=1, T\_MAX=302)