HGHsc/ABINIT

HGHk-sc NCPP dataset / ABINIT 7.10.2

`name and version of the code:` ABINIT 7.10.2
`type of basis set:` plane waves
`method:` norm-conserving pseudopotentials (HGHk-sc)

GENERAL INFORMATION

`exchange-correlation functional`	PBE
`relativistic scheme`	core fully relativistic valence scalar relativistic (Koelling-Harmon)
`assignment of core / valence states`	see table (Z_{val})
`basis set size`	cut-off energy = 250 Ry
`k-mesh density`	see table (k-point mesh in the full 1st Brillouin zone of the conventional cell *kpts*, and number of irreducible k-points # *k*)
`reciprocal-space integration method`	Fermi-Dirac smearing with a fictitious temperature corresponding to 0.01 eV

METHOD-SPECIFIC INFORMATION

none

ADDITIONAL COMMENTS

The ASE script used to generate these data has been included after the table.

REFERENCES

`potentials`

[1] http://cp2k.web.psi.ch/potentials/
[2] S. Goedecker, M. Teter, and J. Hutter, *Phys. Rev. B* **54**, 1703–1710 (1996).
[3] C. Hartwigsen, S. Goedecker, and J. Hutter, *Phys. Rev. B* **58**, 3641–3662 (1998).
[4] M. Krack, *Theor. Chem. Acc.* **114**, 145–152 (2005).

`code`

[5] X. Gonze, J.-M. Beuken, R. Caracas, F. Detraux, M. Fuchs, G.-M. Rignanese, L. Sindic, M. Verstraete, G. Zerah, F. Jollet, M. Torrent, A. Roy, M. Mikami, Ph. Ghosez, J.-Y. Raty and D. C. Allan, *Comput. Mater. Sci.* **25**, 478–492 (2002).
[6] X. Gonze, B. Amadon, P.-M. Anglade, J.-M. Beuken, F. Bottin, P. Boulanger, F. Bruneval, D. Caliste, R. Caracas, M. Côté, T. Deutsch, L. Genovese, Ph. Ghosez, M. Giantomassi, S. Goedecker, D. R. Hamann, P. Hermet, F. Jollet, G. Jomard, S. Leroux, M. Mancini, S. Mazevet, M. J. T. Oliveira, G. Onida, Y. Pouillon, T. Rangel, G.-M. Rignanese, D. Sangalli, R. Shaltaf, M. Torrent, M. J. Verstraete, G. Zerah and J. W. Zwanziger, *Comput. Phys. Commun.* **180**, 2582–2615 (2009).
[7] M. Torrent, F. Jollet, F. Bottin, G. Zérah, X. Gonze, *Comput. Mater. Sci.* **42**, 337–351, (2008).

`scalar relativity`

[8] D. D. Koelling and B. N. Harmon, *J. Phys. C: Solid State* **10**, 3107–3114 (1977).

Table I. Calculation settings and results per element: valence Z_{val}, k-point mesh in the full 1st Brillouin zone of the conventional cell *kpts* and number of irreducible k-points # k, equilibrium volume per atom V_0, bulk modulus B_0, pressure derivative of the bulk modulus B_1.

	Z_{val} [–]	*kpts* [–]	# k [–]	V_0 [Å^3/atom]	B_0 [GPa]	B_1 [–]
H	1	28×28×20	7 840	17.422	10.262	2.683
He	2	40×40×22	8 800	17.781	0.865	6.201
Li	3	38×38×38	27 436	20.210	13.848	3.345
Be	4	52×52×28	18 928	7.893	122.959	3.332
B	3	26×26×24	8 112	7.223	234.668	3.435
C	4	48×48×12	6 912	11.633	207.213	3.551
N	5	16×16×16	176	28.794	53.568	3.666
O	6	26×24×24	3 744	18.681	50.004	3.518
F	7	16×28×14	3 136	19.292	33.769	3.730
Ne	8	22×22×22	286	24.799	1.730	9.142
Na	9	32×32×32	16 384	37.586	7.489	6.691
Mg	10	36×36×20	6 480	23.201	38.235	−12.582
Al	3	24×24×24	364	16.460	77.517	4.861
Si	4	32×32×32	16 384	20.355	88.278	4.278
P	5	30×8×22	1 320	21.260	68.722	4.313
S	6	38×38×38	27 436	17.035	84.313	4.049
Cl	7	12×24×12	864	38.257	19.302	4.372
Ar	8	16×16×16	120	52.181	0.755	7.308
K	9	20×20×20	220	73.602	3.594	3.773
Ca	10	18×18×18	165	42.276	17.618	3.379
Sc	11	34×34×20	5 780	24.630	54.548	3.392
Ti	12	40×40×22	8 800	17.414	111.829	3.603
V	13	34×34×34	969	13.473	182.469	3.911
Cr	14	36×36×36	1 140	12.175	136.321	7.032
Mn	15	28×28×28	2 744	12.006	119.492	5.921
Fe	16	36×36×36	1 140	11.464	177.303	7.785
Co	17	46×46×24	12 696	10.915	210.008	5.081
Ni	18	28×28×28	560	10.925	193.949	4.730
Cu	19	28×28×28	560	11.955	144.750	5.652
Zn	20	44×44×20	9 680	15.234	74.393	4.628
Ga	13	22×12×22	1 452	20.439	46.913	7.184
Ge	4	30×30×30	13 500	24.071	57.597	4.789
As	5	30×30×10	4 500	22.565	68.686	4.278
Se	6	26×26×20	6 760	29.726	47.199	4.445
Br	7	12×24×12	432	39.336	22.570	4.842
Kr	8	16×16×16	120	65.867	0.649	7.221
Rb	9	18×18×18	165	91.058	2.794	3.799
Sr	10	16×16×16	120	54.452	11.304	5.145
Y	11	32×32×18	4 608	32.884	41.228	3.132
Zr	12	36×36×20	6 480	23.353	93.903	3.270
Nb	13	30×30×30	680	18.141	170.147	3.696
Mo	14	32×32×32	816	15.819	259.942	4.350
Tc	15	42×42×22	9 702	14.477	299.499	4.529
Ru	16	42×42×24	10 584	13.802	312.577	4.875
Rh	17	26×26×26	455	14.090	257.333	5.209
Pd	18	26×26×26	455	15.369	169.094	5.544
Ag	19	24×24×24	364	18.022	89.845	6.078
Cd	12	38×38×18	6 498	22.869	44.751	6.972

In	13	30×30×20	1 200	27.664	35.871	5.068
Sn	4	26×26×26	8 788	37.094	34.150	4.716
Sb	5	26×26×8	2 704	31.977	50.220	4.513
Te	6	26×26×16	5 408	35.162	44.717	4.702
I	7	12×22×10	330	50.546	18.567	5.055
Xe	8	14×14×14	84	86.827	0.542	7.182
Cs	9	16×16×16	120	116.830	1.962	3.608
Ba	10	20×20×20	220	63.430	8.747	2.063
Lu	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*
Hf	12	36×36×20	6 480	22.470	107.014	3.374
Ta	13	30×30×30	680	18.151	193.434	3.533
W	14	32×32×32	816	15.999	303.367	4.101
Re	15	42×42×22	9 702	14.818	367.741	4.401
Os	16	42×42×24	10 584	14.137	403.796	4.799
Ir	17	26×26×26	455	14.342	354.960	5.096
Pt	18	26×26×26	455	15.603	253.682	5.477
Au	19	24×24×24	364	17.888	143.464	6.011
Hg	12	24×24×28	1 092	29.405	8.379	9.825
Tl	13	32×32×18	4 608	31.484	27.024	5.472
Pb	14	20×20×20	220	32.047	39.979	5.605
Bi	15	26×26×8	2 704	36.970	42.722	4.668
Po	6	30×30×30	680	37.627	45.541	5.012
Rn	8	14×14×14	84	93.350	0.539	7.197

HGHsc/ABINIT

HGHk-sc NCPP dataset / ABINIT 7.10.2

script for the Atomic Simulation Environment (ASE)
S. R. Bahn, K. W. Jacobsen, *Comput. Sci. Eng.* **4**, 56–66 (2002)
https://wiki.fysik.dtu.dk/ase/

```
import os
import sys
import time

import numpy as np

import ase.db
from ase.units import Rydberg
from ase.utils import opencew
from ase.calculators.calculator import kpts2mp
from ase.io.trajectory import PickleTrajectory
from ase.calculators.abinit import Abinit
from ase.test.tasks.dcdft import DeltaCodesDFTCollection as Collection

collection = Collection()

if len(sys.argv) == 1:
    names = collection.names
else:
    names = [sys.argv[1]]

c = ase.db.connect('dcdft_abinit_hgh.db')

ecut = 250

kptdensity = 16.0  # this is converged
width = 0.01
ecutsm = 0.0
fband = 1.5
tolsym = 1.e-12

linspace = (0.98, 1.02, 5)  # eos numpy's linspace
linspacestr = ''.join([str(t) + 'x' for t in linspace])[:-1]

code = 'abinit' + '-' + '_c' + str(ecut) + '_e' + linspacestr
code = (code + '_k' + str(kptdensity) + '_w' + str(width) + '_s' + str(ecutsm) +
       '_t' + str(tolsym))

for name in names:
    # save all steps in one traj file in addition to the database
    # we should only used the database c.reserve, but here
    # traj file is used as another lock ...
    fd = opencew(name + '_' + code + '.traj')
    if fd is None:
        continue
    traj = PickleTrajectory(name + '_' + code + '.traj', 'w')
    atoms = collection[name]
    if name == 'Mn':  # fails to find the right magnetic state
        atoms.set_initial_magnetic_moments([10., 20., -10., -20.])
```

```
if name == 'Co':  # fails to find the right magnetic state
    atoms.set_initial_magnetic_moments([10., 10.])
if name == 'Ni':  # fails to find the right magnetic state
    atoms.set_initial_magnetic_moments([10., 10., 10., 10.])
cell = atoms.get_cell()
kpts = tuple(kpts2mp(atoms, kptdensity, even=True))
kwargs = {}
# loop over EOS linspace
for n, x in enumerate(np.linspace(linspace[0], linspace[1], linspace[2])):
    id = c.reserve(name=name, ecut=ecut,
                   linspacestr=linspacestr,
                   kptdensity=kptdensity, width=width, ecutsm=ecutsm,
                   fband=fband, tolsym=tolsym,
                   x=x)
    if id is None:
        continue
    # perform EOS step
    atoms.set_cell(cell * x, scale_atoms=True)
    # set calculator
    atoms.calc = Abinit(
        pps='hgh.k',  # uses highest valence hgh.k pps
        label=name + '_' + code + '_' + str(n),
        xc='PBE',
        kpts=kpts,
        ecut=ecut*Rydberg,
        occopt=3,
        tsmear=width,
        ecutsm=ecutsm,
        toldfe=1.0e-6,
        nstep=900,
        pawovlp=-1,  # bypass overlap check
        fband=fband,
        # http://forum.abinit.org/viewtopic.php?f=8&t=35
        chksymbreak=0,
        tolsym=tolsym,
        prtwf=0,
        prtden=0,
    )
    atoms.calc.set(**kwargs)  # remaining calc keywords
    t = time.time()
    atoms.get_potential_energy()
    c.write(atoms,
            name=name, ecut=ecut,
            linspacestr=linspacestr,
            kptdensity=kptdensity, width=width, ecutsm=ecutsm,
            fband=fband, tolsym=tolsym,
            x=x,
            time=time.time()-t)
    traj.write(atoms)
    wfk = name + '_' + code + '_' + str(n) + 'o_WFK'
    if os.path.exists(wfk): os.remove(wfk)
    del c[id]
```