things_to_do

November 20, 2018

Contents

1 Tasks

1.1 TODO Finish doing the gamma surfaces for all planes for pure titanium.

1.1.1 Checking the convergence criteria

- Now checking the convergence criteria.
- Going to check:
  - How the lattice parameters change with the fineness of the k mesh
    - Maybe with a less fine k mesh the lattice parameters become awful
  - SOLUTION: The lattice parameters do not change that much under differences with the k mesh. File with change of the lattice parameters with k mesh.
  - a_{vsnk}
  - c_{vsnk}
  - e_{vsnk}
  - How does rmaxh change the energy of a supercell
  - How does the number of neighbours change and what is the relation between rmaxh and larger cell sizes.

1.1.2 DONE Implement Homogenous Shear boundary conditions for gamma surface calculation.

1.2 TODO Python script: remove include statements –> One file.

1.3 TODO Summarise UCL DFT lectures.

1.4 TODO Write first paragraph of Literature review

1.4.1 TODO Summarise Stacking Faults and write review

1.4.2 TODO Write up the tight binding fitting of oxygen and an explanation for paramagnetism.

1.4.3 TODO Summarise dislocations and Oxygen interactions (review)

1.5 TODO Write summary of org-mode

1.6 DONE Look at the range of the bond integrals we have in Titanium graphically.

1.6.1 Pair potentials in tbe code

- Pair potential is constructed by makvpp.f.
- This calls vppder.f which actually evaluates the pair potential at that point
- In makvpp.f, if in the range of $r_1 < r < r_c$, then augmentative/multiplicative polynomial is used.
  - To make this polynomial pcut45.f is used.
  - Depending on the degree of polynomial we have this structure:

```
      rr = r1 - r2
      xr1 = x - r1
      xr2 = x - r2

      c = val*rr*rr
      if (n == 5) then
pnorm = rr**(-5)
a = (0.5d0*curv*rr - 3d0*slo)*rr + 6d0*val
b = (slo*rr - 3d0*val)*rr
      elseif (n == 4) then
```

```
pnorm = rr**(-4)
a = (0.5d0*curv*rr - 2d0*slo)*rr + 3d0*val
b = (slo*rr - 2d0*val)*rr
      p2 = pnorm*(c + xr1*(b + xr1*a))
      dp2 = pnorm*(b + xr1*2d0*a)
      ddp2 = pnorm*2d0*a
      e = p2 * xr2**(n-2)
      de = (xr2*dp2 + float(n-2)*p2) * xr2**(n-3)
      dde = (xr2*xr2*ddp2+float(2*(n-2))*xr2*dp2+float((n-2)*(n-3))*p2)
C ... e, de and dde are the values and derivatives of the polynomial in the r
```

- So the form of the polynomial used is
  - $$P_5(x) = (x - r_2)^3 P_2(x)$$
  - $$P_2(x) = a(x - r1)^2 + b(x - r_1) + c$$
  - $$a = \frac{1}{(r1 - r2)^5}\{\frac{1}{2}(r_1 - r_2)^2 f"(r_1) - 3(r_1 - r_2)f'(r_1) + 6f(r_1)\}$$
  - $$b = \frac{1}{(r_1 - r_2)^4}\{f'(r_1) * (r_1 - r_2) - 3f(r_1)\}$$
  - $$\frac{1}{(r_1 - r_2)^5}x$$
  - $$c = \frac{f(r_1)}{(r_1 - r_2)^3}$$
  - Where $f(x)$ is the function that needs to be cut

- Current model has this

```
Ti,Ti:
   type 2 (Exp. decay), V(d) = a exp (- b d)
     sss    sps    pps    ppp    sds    pds    pdp    dds    ddp    ddd
   coeff:   0.00   0.00   0.00   0.00   0.00   0.00   0.00  -2.75   1.84  -0.46
   decay:   0.00   0.00   0.00   0.00   0.00   0.00   0.00   0.71   0.71   0.71
   cutoff type 2 (multiplicative), 5th order polynomial, range [r1, rc]
     sss    sps    pps    ppp    sds    pds    pdp    dds    ddp    ddd
   r1:      0.00   0.00   0.00   0.00   0.00   0.00   0.00   6.20   6.20   6.20
   rc:      0.00   0.00   0.00   0.00   0.00   0.00   0.00   8.50   8.50   8.
```

1.6.2 Bond integrals from tbe

- So bond integrals from titanium look like this, from this file $\text{plot}_{\text{bondintegrals}}$.py

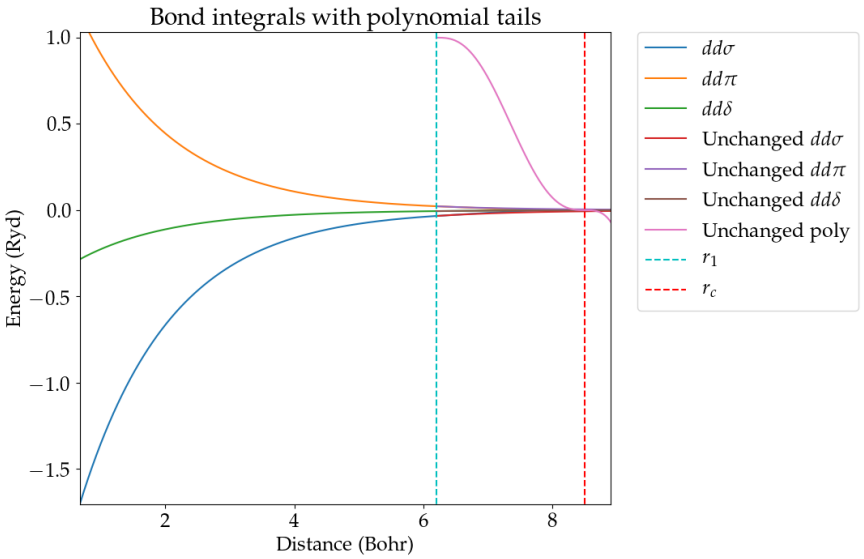

Figure 1: Bond integrals with multiplicative polynomial cutoffs.

1.7 DONE Investigate why rmaxh changes energy

- Variation of rmaxh does not change the energy
- Obviously the number of neighbours changes with rmaxh.
- Conclusion: rmaxh only determines what atoms are its neighbours.
- This is the file which investigates this: $\text{check}_{\text{rmaxhenergynumberneighbours}}$
- Here is the data: Energy data for energy vs rmaxh rmaxh data for energy/$\text{n}_{\text{neighbours}}$ vs rmaxh $\text{n}_{\text{neighbours}}$ for $\text{n}_{\text{neighbours}}$ vs rmaxh
- The output pictures are this:

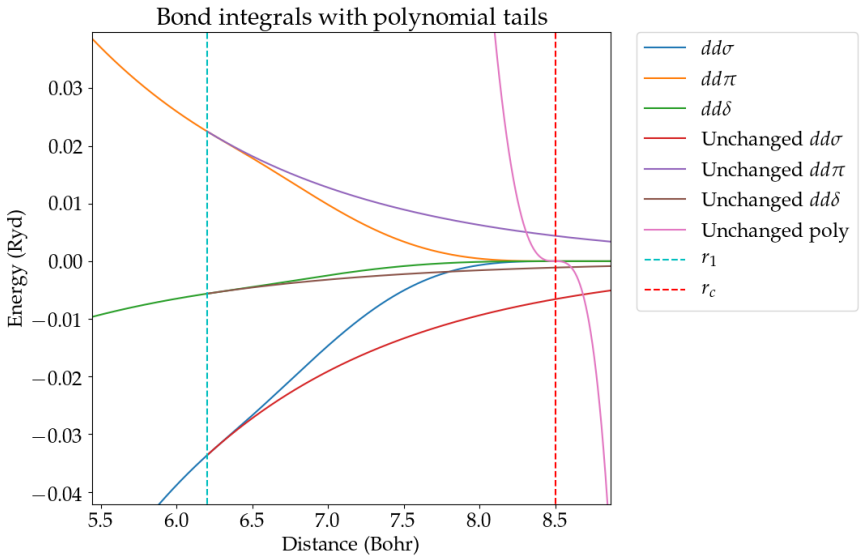

Figure 2: Bond integrals with multiplicative polynomial cutoffs: zoomed in.

1.8 DONE Show supercell of BOP working

2 General notes of codes.

2.1 Pair potentials in tbe code

- Pair potential is constructed by makvpp.f.
- This calls vppder.f which actually evaluates the pair potential at that point
- In makvpp.f, if in the range of $r_1 < r < r_c$, then augmentative/multiplicative polynomial is used.
  - To make this polynomial pcut45.f is used.
  - Depending on the degree of polynomial we have this structure:

```
rr = r1 - r2
xr1 = x - r1
xr2 = x - r2

c = val*rr*rr
if (n == 5) then
```

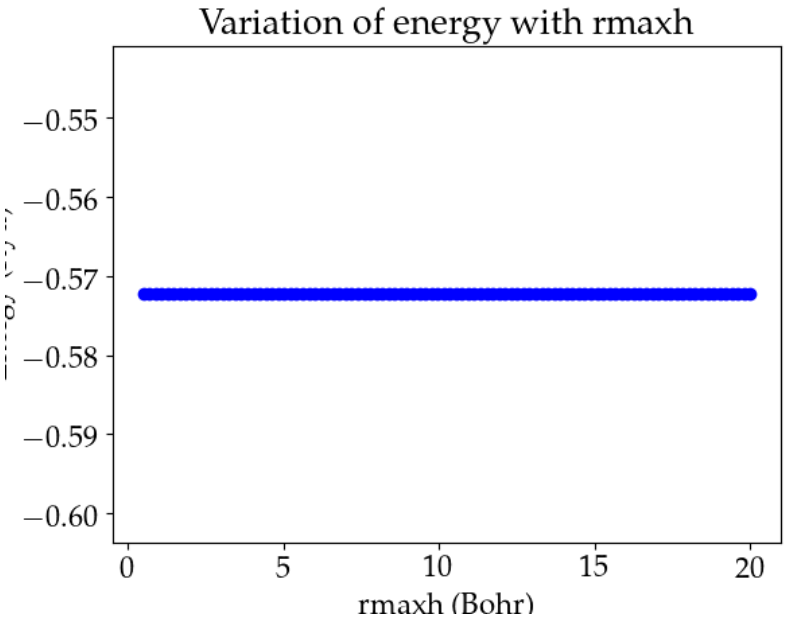

Figure 3: Variation of energy with change in rmaxh

```
pnorm = rr**(-5)
a = (0.5d0*curv*rr - 3d0*slo)*rr + 6d0*val
b = (slo*rr - 3d0*val)*rr
      elseif (n == 4) then
pnorm = rr**(-4)
a = (0.5d0*curv*rr - 2d0*slo)*rr + 3d0*val
b = (slo*rr - 2d0*val)*rr
      p2 = pnorm*(c + xr1*(b + xr1*a))
      dp2 = pnorm*(b + xr1*2d0*a)
      ddp2 = pnorm*2d0*a
      e = p2 * xr2**(n-2)
      de = (xr2*dp2 + float(n-2)*p2) * xr2**(n-3)
      dde = (xr2*xr2*ddp2+float(2*(n-2))*xr2*dp2+float((n-2)*(n-3))*p2)
C ... e, de and dde are the values and derivatives of the polynomial in the r
```

– So the form of the polynomial used is

  *

$$P_5(x) = (x - r_2)^3 P_2(x)$$

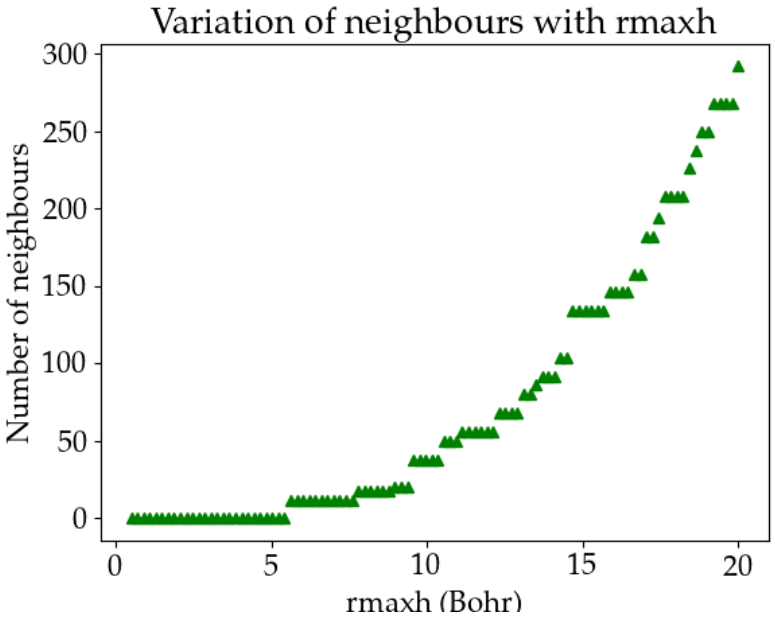

Figure 4: Variation of number of neighbours with change in rmaxh

* $$P_2(x) = a(x - r1)^2 + b(x - r_1) + c$$

* $$a = \frac{1}{(r1 - r2)^5}\{\frac{1}{2}(r_1 - r_2)^2 f"(r_1) - 3(r_1 - r_2)f'(r_1) + 6f(r_1)\}$$

* $$b = \frac{1}{(r_1 - r_2)^4}\{f'(r_1) * (r_1 - r_2) - 3f(r_1)\}$$

* $$\frac{1}{(r_1 - r_2)^5}x$$

* $$c = \frac{f(r_1)}{(r_1 - r_2)^3}$$

  * Where $f(x)$ is the function that needs to be cut

- Current model has this

```
Ti,Ti:
   type 2 (Exp. decay), V(d) = a exp (- b d)
     sss    sps    pps    ppp    sds    pds    pdp    dds    ddp    ddd
   coeff:   0.00   0.00   0.00   0.00   0.00   0.00   0.00  -2.75   1.84  -0.46
   decay:   0.00   0.00   0.00   0.00   0.00   0.00   0.00   0.71   0.71   0.71
   cutoff type 2 (multiplicative), 5th order polynomial, range [r1, rc]
     sss    sps    pps    ppp    sds    pds    pdp    dds    ddp    ddd
   r1:      0.00   0.00   0.00   0.00   0.00   0.00   0.00   6.20   6.20   6.20
   rc:      0.00   0.00   0.00   0.00   0.00   0.00   0.00   8.50   8.50   8.
```

- So bond integrals from titanium look like this
- Bond integrals with multiplicative polynomial cutoffs.
- Bond integrals with multiplicative polynomial cutoffs: zoomed in.

2.2 Notes for the gamma surfaces

- Seems like some atoms are missing in the site file when it is being read in to tbe
- This means that there are some erroneous forces that make the program exit.
  - SOLUTION: Coordinates were not in units of alat.

2.2.1 Relaxing in tbe

- To relax in tbe need to modify:
  - Ewald tolerance: ewtol
    - This can generally be set quite low: 1d-14
  - Convergence criteria:
    - gtol: The tolerance in the force for convergence e.g. 1d-8
    - xtol: The tolerance in the atomic positon e.g. 1d-8.

2.3 Ti Swarm fitting.

- Here used fitting with uniform weights across all target quantities without a regularisation of the parameters.

- It can be seen that the lattice parameters aren't as good as they could be. This calls for the use of weighted parameters.
- Have now started weighted parameter search for the best parameters with regards to titanium.

```
Build Objective Function
 ...with L1 norm
| Quantity      |  predicted |     target | squared diff. |    p_norm |   weight |  obj
|---------------+------------+------------+---------------+-----------+----------+-----
| a_hcp:        |   4.744693 |   5.576790 |      0.692385 |  0.832097 | 1.000000 |   1
| c_hcp:        |   7.495518 |   8.852101 |      1.840316 |  1.356583 | 1.000000 |   3
| c_11:         | 174.924630 | 176.100000 |      1.381495 |  1.175370 | 1.000000 |   2
| c_33:         | 190.161490 | 190.500000 |      0.114589 |  0.338510 | 1.000000 |   0
| c_44:         |  54.517320 |  50.800000 |     13.818465 |  3.717320 | 1.000000 |  17
| c_12:         |  65.010403 |  86.900000 |    479.154446 | 21.889597 | 1.000000 | 501
| c_13:         |  73.335501 |  68.300000 |     25.356271 |  5.035501 | 1.000000 |  30
| a_omega:      |   7.331279 |   8.732543 |      1.963543 |  1.401265 | 1.000000 |   3
| c_omega:      |   4.768459 |   5.323431 |      0.307994 |  0.554972 | 1.000000 |   0
| u_omega:      |   1.000025 |   1.000000 |      0.000000 |  0.000025 | 1.000000 |   0
| DeltaE_O_hcp: |  -1.170318 |  -0.734754 |      0.189716 |  0.435564 | 1.000000 |   0
| a_bcc:        |   5.331467 |   6.179489 |      0.719140 |  0.848021 | 1.000000 |   1
| bandwidth:    |   0.325300 |   0.426000 |      0.010140 |  0.100700 | 1.000000 |   0
Objective function: 563

Objective Function = 563.2340263379571
Stopping search: Swarm best position change less than 1e-08
[ 0.34606728 -0.22330935 65.79555644 0.52284417 0. -0.62229341 1.98315066]
563.2340263379571
```

3 Useful Notes

3.1 Org-mode

```
(setq org-latex-create-formula-image-program 'dvipng)
```

3.2 Physics

3.2.1 Hartree-Fock

- Hartree-Fock is a method of calculating the energy of a configuration with exact exchange.

- This is done by essentially putting everything we don't know into the kinetic energy functional.
- Hamiltonian is split into contributions:
  - $$\hat{H} = \hat{T} + \hat{V}_{\text{ext}} + \hat{G}$$
  - $\hat{G} = \hat{J} - \hat{K}$
  - $\hat{J}$ is the coulombic interaction:
  - $$\langle \mathbf{r} | \, \hat{J} \, | \mathbf{n} \rangle = \int \frac{\langle \mathbf{r} | n \rangle}{|\mathbf{r} - \mathbf{r}'|} d\mathbf{r}$$
  - So $$E_{\text{H}} = \int \frac{\rho\mathbf{r}\rho\mathbf{r}'}{|\mathbf{r} - \mathbf{r}'|}$$
  - This includes fictitious self-interaction of electron density.
  - The Exchange functional removes this part, thus lowering the energy
- This method is used in Hybrid DFT. This corrects band gaps mainly. But there are also problems.

4 DFT Lectures UCL

4.1 David Bowler O(N) DFT

4.1.1 Types of Exchange-correlation Functionals

1. LDA
   - The electron density is the same as a uniform electron gas.
   - Exchange is Slater.
   - Still parameterised (Ceperly). Parameters from Quantum Monte-Carlo calculations.
2. GGA
   - The gradient of the electron density is included in functional.

- Have the reduced density

$$\frac{\nabla n(\mathbf{r})}{n(\mathbf{r})}$$

.

(a) Perdew-Burke-Ernzerhof

- $$E_{\mathrm{x}} = \int n(\mathbf{r})\epsilon_{\mathrm{xc}}[n(\mathbf{r})]F_{\mathrm{x}}(S)d\mathbf{r}$$
- $$E_{\mathrm{c}} = \int n[\epsilon_{\mathrm{c}} + H(n,S)]d\mathbf{r}$$
- These integrals are then fitted to various limits.

3. Hybrid Functionals

- These are functionals to correct the self-interaction energy that is apparent in the previously mentioned functionals.
- The Hartree term
$$V_{\mathrm{H}} = \int \frac{\rho(\mathbf{r})}{|\mathbf{r}-\mathbf{r}'|}d\mathbf{r}$$
- The exchange term cancels the celf interaction.
- Generally only a part of this Hartree-Fock calculation is included in the function otherwise it is not stable.

DFT speed is limited by how it can find the energies of the system we are interested in. Diagonalisation is inherently an $\mathcal{O}(N^3)$ process.

To actually build the hamiltonian it is of $\mathcal{O}(N^2)$. Solving is $\mathcal{O}(N^3)$.

How do we solve for DFT? Generally it depends on the choice of functional we have. Hybrid functionals almost scale as $\mathcal{O}(N^4)$ due to the inclusion of exact exchange interaction by Hartree-Fock. Because of this exact exchange, there are better band gaps .

The $\mathcal{O}(N)$ DFT generally comes because of the manipulation of sparse matrices. Insead of matrix multiplication being of $\mathcal{O}(N^3)$ we can have matrix multiplication being of $\mathcal{O}(N)$.

The reason we can essentially do $\mathcal{O}(N)$ is that in the Kohn-Sham equations, the density is actually a local function ($n(\mathbf{r})$, not $n(\mathbf{r}-\mathbf{r}')$) This means that in theory we can actually have a theory which sufficiently

describes the dynamics of a given system with an electron density that is local in space. In many DFT codes however, the electron density is non-local ($n(\mathbf{r}-\mathbf{r}')$), and this slows down the calculation. To actually make it $\mathcal{O}(N)$, we have to have range cutoffs for the interactions of the atoms. This means that the hamilitonian is sparse as quite a lot of the elements are zero such that we can use methods that involve $\mathcal{O}(N)$ multiplication.

When it comes to Structural relaxation there are a few things that come to mind when structures are not converging: there is usually only one atom that has some huge force on it. Consider the boundary conditions.

For faster diagonalisation of the hamiltonian matrix it may be useful to look at methods such as Krylov-Subsapace, Lanczos and folded-spectrum methods.

4.2 Jochen Blumberger: Molecular dynamics

- Molecular dynamics is important. (Even at 0K there is a zero point energy of vibration).
- Need theory to see how atoms move

4.2.1 Born-Oppenheimer approximation

- Have hamiltonian that consists of interaction between:
  - nucleus-nucleus
  - nucleus-electron
  - electron-electron
- First assumption is that we can write the eigenfunction of this large hamiltonian as a product state consisting of an electronic ground state and nuclear eigenstate.
- Second approximation is that we are able to say, as the mass of the ion $M_I \sim 1000 m_e$ then we can say that the kinetic energy term of with regard to the nucleus positions will be small.
- From this we can say that the action of this nuclear kinetic energy operator on the electronic eigenstate is small.

- This means we can neglect the **electronic** wavefunction, and work with the equation

$$\hat{H}\Phi(\mathbf{R}) = E^0_{\mathbf{R}}\Phi(\mathbf{R})$$

  - Where $E^0_{\mathbf{R}}$ is the ground state energy hypersurface from the electronic wavefunction. We get this from DFT calculations.

- Even now we can only really calculate 8 degrees of freedom for the Nuclear wavefunction.

4.2.2 Molecular Dynamics

1. Verlet Algorithm

   - This algorithm simply uses the forward and backward derivative of the nuclear positions and adds them together to get a formula for the positon.

     –

$$\mathbf{R}_I(t+\delta t) = 2\mathbf{R}_I - \mathbf{R}_I(t-\delta t) + \frac{f_I(t)}{M_I}\delta t^3\mathcal{O}(\delta t^4)$$

     –

$$\dot{\mathbf{R}}_I(t) = \frac{1}{2\delta t}[\mathbf{R}_I(t+\delta t) - \mathbf{R}_I(t-\delta t)] + \mathcal{O}(\delta t^3)$$

     – This causes a problem however: the velocity is calculated a step after that of the positons. So this leads to the Velocity Verlet algorithm.

2. Velocity Verlet Algorithm

   - For this algorithm the forward derivative with respect to nuclear positions is used with a calculation of the force at a later time.
   - Then the taylor expansion of the position at time t is used with the terms of later time.
   -

$$\mathbf{R}_I(t+\delta t) = \mathbf{R}_I(t) + \dot{\mathbf{R}}_I\delta t + \frac{f_I(t)}{M_I}\delta t^3 + \mathcal{O}(\delta t^3)$$

   -

$$\dot{\mathbf{R}}_I(t+\delta t) = \dot{\mathbf{R}}_I(t) + \frac{1}{2M_I}[\mathbf{f}_I(t+\delta t) + \mathbf{R}_I(t)] + \mathcal{O}(\delta t^3)$$

3. How to calculate the forces

   - Use the Hellmann-Feynman theorem.

–

$$\mathbf{f}_I = < \psi^0_{\mathbf{R}} | \frac{\partial}{\partial \mathbf{R}_I} \hat{H} | \psi^0_{\mathbf{R}} >$$

5 org-mode cheat sheet

- New TODO: M-<shift>-<ret>
- Done TODO: C-c C-t
- Links: [[]][] [link] then [description]
- Open link: Move over cursor and do C-c C-o
- Link to local files:
  - Open file (C-x C-f) then do C-c l,
  - then go back to org file and do C-c C-l (e.g. $\text{Upgrade}_{\text{reportplusnotes}}$)
- To remove window in buffer C-x 0
- Overview of document <shift>-<TAB> to condense to titles.
- Can have global todo list
- < s TAB expands to a 'src' code block.
- < l TAB expands to:
- If I want more help I can go to the org-mode manual