FORTRAN 90 Ornstein-Zernike solver :: version 1.12

Patrick B Warren, Unilever R&D Port Sunlight (February 2018)

 If you end up publishing results based on this, please cite

> P. B. Warren, A. Vlasov, L. Anton and A. J. Masters "Screening properties of Gaussian electrolyte models, with application to dissipative particle dynamics", J. Chem. Phys. **138**, 204907 (2013).

Features of version 1.12:

- modern FORTRAN 90 based, with example `python` driver scripts;
- HNC, MSA, RPA, and EXP closures;
- fast Ng solver, and Ng decomposition for electrostatics;
- multicomponent (arbitrary number of components);
- hard core (RPM-like) and soft core (DPD-like) potentials;
- thermodynamics computed (full HNC free energy added in v1.12);
- fully open source.

Versions 1.10, 1.11 fixed bugs: the contact virial pressure contribution was missing from the HNC free energy, and the compressibility and HNC chemical potentials were missing a mean-field term for the off-SYM RPM potential. Added in version 1.7–1.9: MSA closure and EXP; exact and approximate pair potentials for Slater smearing; EXP for hard core potentials fixed, new examples added; source code brought up to a modern FORTRAN 90 standard.

Contents

1 Introduction

The FORTRAN 90 module `oz_mod.f90` implements an Ornstein-Zernike solver for multicomponent mixtures of possibly charged particles, returning both structural and thermodynamic information. The closures currently implemented are the HNC, MSA, RPA, and EXP. Though we say it ourselves, the code is *fast* since it is written in native FORTRAN and implements the Ng acceleration schemes [2]. The interaction potentials are specified in their own routines and the code could be used for other potentials with minor modifications. Further technical background to liquid state integral equation methods can be found in Hansen and McDonald [3], Kelley and Montgomery Pettitt [4], and Vrbka *et al.* [5]. We note that Vrbka provides an independent package called 'pyOZ', implemented entirely in `python` [5]. There are some differences with the present approach: the Ornstein-Zernike relation is normalised slightly differently, a conjugate gradient method is used to accelerate convergence, and pyOZ provides other closures in addition to the ones provided here. The present code is based on an HNC code developed by Lucian Anton, modified for dissipative particle dynamics (DPD) by Andrey Vlasov with help from Lucian and Andrew Masters. The code was later converted to use the open source libraries FFTW and LAPACK by Patrick Warren, and rewritten to be compatible with `python` using the `f2py` interface generator.

2 Theoretical background

2.1 Single component HNC

We suppose that the interaction between a pair of particles is given by the potential $v(r)$. We introduce the following quantities [3]: the pair distribution function $g(r)$, the total correlation function $h(r) \equiv g(r) - 1$, the direct correlation function $c(r)$ defined by the Ornstein-Zernike relation below, and the indirect correlation function $e(r) \equiv h(r) - c(r)$. Note that the indirect correlation function is called $\gamma(r)$ by Vrbka *et al.* [5] and $b(r)$ by Hansen and McDonald [3]. We also introduce the Fourier transforms, *e. g.*

$$\tilde{h}(k) = \int d^3\mathbf{r}\, e^{-i\mathbf{k}\cdot\mathbf{r}}\, h(r)\,, \tag{1}$$

which simplifies to the Fourier-Bessel transform

$$\tilde{h}(k) = \frac{4\pi}{k} \int_0^\infty dr\, \sin(kr)\, r\, h(r)\,, \tag{2}$$

and

$$h(r) = \int \frac{d^3\mathbf{k}}{(2\pi)^3}\, e^{i\mathbf{k}\cdot\mathbf{r}}\, \tilde{h}(k)\,, \tag{3}$$

which simplifies to

$$h(r) = \frac{1}{2\pi^2 r} \int_0^\infty dk\, \sin(kr)\, k\, \tilde{h}(k)\,. \tag{4}$$

In terms of the Fourier transforms the Ornstein-Zernike (OZ) relation is

$$\tilde{h}(k) = \tilde{c}(k) + \rho\, \tilde{h}(k)\, \tilde{c}(k) \tag{5}$$

where ρ is the density. This confirms the relevance of the standard choice of normalisation of the 3d Fourier transform pair which puts the factor $1/(2\pi)^3$ into the back transform. The OZ relation can be rearranged to

$$\tilde{h}(k) = \frac{\tilde{c}(k)}{1 - \rho\tilde{c}(k)} \tag{6}$$

(*c. f.* Eq. (3.5.13) in Ref. [3]). The hyper-netted chain (HNC) closure is defined in real space and is

$$g(r) = \exp[-\beta v(r) + h(r) - c(r)] \tag{7}$$

(*c. f.* Eq. (4.3.19) in Ref. [3]) where $\beta = 1/k_{\mathrm{B}}T$. It amounts the neglect of the bridge diagrams. For hard spheres HNC is known to be inferior to Percus-Yevick, but for soft potentials like DPD it generally gives excellent results. In the presence of hard cores, we note that $\exp[-\beta v]$ is still well defined (and zero, in fact) where βv is infinite. We can accommodate hard cores therefore by making sure that we always work with $\exp[-\beta v]$ instead of βv in the numerics. The function $\exp[-\beta v]$ can be pre-computed for a given potential.

To obtain the algorithm implemented in the code we rewrite Eqs. (6) and (7) in terms of $c(r)$ and $e(r)$. The OZ relation becomes

$$\tilde{e} = \frac{\tilde{c}}{1 - \rho\tilde{c}} - \tilde{c} \tag{8}$$

and the HNC closure becomes

$$c = \exp[-\beta v + e] - e - 1\,. \tag{9}$$

The algorithm is as follows. We start with some guess for the direct correlation function $c(r)$. We Fourier transform this to $\tilde{c}(k)$ and solve the OZ

relation in Eq. (8) to get the indirect correlation function (in reciprocal space) $\tilde{e}(k)$. We Fourier back-transform to get $e(r)$ and plug this into the HNC closure in the form of Eq. (9) to get a new direct correlation function $c(r)$. This cycle is iterated until $c(r)$ and $e(r)$ converge to a self-consistent solution pair. A direct approach like this is usually numerically unstable so in the Picard scheme we mix a little of the new $c(r)$ into the old $c(r)$, and iterate until we converge to a solution. In the Ng acceleration scheme we keep track of the convergence trajectory and use this to accelerate convergence to the solution. The details of the Ng scheme will not be described here, rather we point the interested reader to Ref. [2] and the code itself. The initial trajectory for the Ng scheme is generated by a few Picard iterations. Some possible initial choices for the direct correlation function $c(r)$ are zero, $-\beta v(r)$ and $\exp[-\beta v(r)] - 1$. For soft potentials any of these will do in principle but the initial rate of convergence may differ. For hard core potentials, the middle option cannot be used.

Now we describe Ng's splitting method for handling long range forces. We partition the interaction potential into a short range part and a long range part,

$$v(r) = v'(r) + v^{\mathrm{L}}(r)\,. \tag{10}$$

It is generally accepted that $c(r) \sim -\beta v^{\mathrm{L}}(r)$ for $r \to \infty$ so we make this explicit by partitioning both the direct and indirect correlation functions,

$$c(r) = c'(r) - \beta v^{\mathrm{L}}(r)\,, \quad e(r) = e'(r) + \beta v^{\mathrm{L}}(r)\,. \tag{11}$$

These are taken to provide the definitions of the offset functions $c'(r)$ and $e'(r)$. We rewrite the OZ relation and the HNC closure in terms of these,

$$\tilde{e}' = \frac{\tilde{c}' - \beta\tilde{v}^{\mathrm{L}}}{1 - \rho(\tilde{c}' - \beta\tilde{v}^{\mathrm{L}})} - \tilde{c}' \tag{12}$$

and

$$c' = \exp[-\beta v' + e'] - e' - 1\,. \tag{13}$$

The first of these requires that we know the Fourier transform of the long range part of the potential. The second requires that we know the exponentiated short range potential. Any hard core interaction can be parked safely in the latter but the consequential freedom to specify $v^{\mathrm{L}}(r)$ inside the hard core should be used wisely!

The main advantage of the Ng split is that $c'(r)$ and $e'(r)$ are now genuinely short-ranged so the numerical behaviour is much better, certain thermodynamic integrals are guaranteed convergence, and the expected asymptotic behaviour of $c(r)$ is explicitly incorporated. The numerical solution scheme based on Eqs. (12) and (13) goes through as before. The initial choices for $c'(r)$ are replicated from above with v replaced by v'.

2.2 Multicomponent HNC

Now we turn to the multicomponent problem. The pair functions become for example $g_{\mu\nu}(r)$, *et c.*, and the problem is specified by the interaction potential $v_{\mu\nu}(r)$ and the species densities ρ_μ. The total density is $\rho = \sum_\mu \rho_\mu$ and the species mole fractions are $x_\mu = \rho_\mu/\rho$. We again split the potential into short and long range contributions,

$$v_{\mu\nu}(r) = v'_{\mu\nu}(r) + v^{\rm L}_{\mu\nu}(r) \tag{14}$$

where typically $v^{\rm L}_{\mu\nu}(r)$ incorporates the long range part of the charge interactions. Often the long range part satisfies a symmetry condition (SYM) where

$$v^{\rm L}_{\mu\nu}(r) = z_\mu z_\nu v^{\rm L}(r) \quad \text{(SYM)} \tag{15}$$

in which z_μ is the valency and $v^{\rm L}(r)$ is the interaction potential between unit charges of the same sign. However we will not assume that $v^{\rm L}_{\mu\nu}(r)$ necessarily meets the SYM condition. As above we define $c'_{\mu\nu} = c_{\mu\nu} + \beta v^{\rm L}_{\mu\nu}$ and $e'_{\mu\nu} = e_{\mu\nu} - \beta v^{\rm L}_{\mu\nu}$. The multicomponent HNC closure becomes

$$c'_{\mu\nu} = \exp[-\beta v'_{\mu\nu} + e'_{\mu\nu}] - e'_{\mu\nu} - 1\,. \tag{16}$$

To derive the OZ relation in a usable form we start from the multicomponent analogue of Eq. (5),

$$\tilde{h}_{\mu\nu} = \tilde{c}_{\mu\nu} + \rho \textstyle\sum_\lambda x_\lambda\, \tilde{c}_{\mu\lambda}\, \tilde{h}_{\lambda\nu}\,. \tag{17}$$

We write this as a matrix relation, introducing R as a diagonal matrix with entries $\rho_\mu = \rho x_\mu$,

$$\tilde{H} = \tilde{C} + \tilde{C}\cdot R\cdot \tilde{H}\,. \tag{18}$$

Note that all the matrices in this are symmetric, so we can equally well write this as

$$\tilde{H} = \tilde{C} + \tilde{H}\cdot R\cdot \tilde{C}\,. \tag{19}$$

Introducing next $\tilde{C}' = \tilde{C} + \beta\tilde{U}^{\rm L}$ and $\tilde{E}' = \tilde{E} - \beta\tilde{U}^{\rm L}$ in Eq. (18), and rearranging, gives eventually

$$\tilde{E}' = [I - (\tilde{C}' - \beta\tilde{U}^{\rm L})\cdot R]^{-1}\cdot[(\tilde{C}' - \beta\tilde{U}^{\rm L})\cdot R\cdot\tilde{C}' - \beta\tilde{U}^{\rm L}]\,. \tag{20}$$

The solution scheme is essentially as before. Given an initial guess for $c'_{\mu\nu}(r)$ we Fourier transform, evaluate the matrix expression in Eq. (20), and Fourier back-transform to obtain $e'_{\mu\nu}(r)$. We substitute this into the HNC closure in Eq. (16) to obtain a new guess for $c'_{\mu\nu}(r)$. The cycle is iterated to convergence and in practice a few Picard iterations are used to pump-prime a multicomponent version of the Ng acceleration scheme. As before, any hard core part of the potential should be parked in $\exp[-\beta v'_{\mu\nu}]$.

2.3 Multicomponent MSA

The closure in the mean spherical approximation (MSA) is $g_{\mu\nu}(r) = 0$ for $r \leq d_{\mu\nu}$, and $c_{\mu\nu}(r) = -\beta v_{\mu\nu}(r)$ for $r > d_{\mu\nu}$. Here, $d_{\mu\nu}$ is the range of the hard core repulsion between species μ and ν (*i. e.* $v'_{\mu\nu} = \infty$ for $r \leq d_{\mu\nu}$). Rewriting as usual in terms of our offset functions the MSA closure becomes

$$c'_{\mu\nu} = \begin{cases} -e'_{\mu\nu} - 1 & (r < d_{\mu\nu})\,, \\ -\beta v'_{\mu\nu} & (r > d_{\mu\nu})\,. \end{cases} \tag{21}$$

This is almost a drop-in replacement for the HNC closure in Eq. (16). For a cold start we can initialise $c'_{\mu\nu} = -1$ for $r \leq d_{\mu\nu}$, and we leave $c'_{\mu\nu} = -\beta v'_{\mu\nu}$ untouched for $r > d_{\mu\nu}$ during the iterative solution. We make sure that whenever the potential is changed $c'_{\mu\nu}$ for $r > d_{\mu\nu}$ is correctly set.

2.4 Multicomponent RPA

The RPA closure is a special case of the MSA when there are no hard cores, and is represented by $c_{\mu\nu}(r) = -\beta v_{\mu\nu}(r)$. This is in fact one of the choices for initialising the HNC solver. Given the HNC machinery, the implementation of the RPA is trivial and comprises a single round trip through the OZ solver starting from $c'_{\mu\nu}(r) = -\beta v'_{\mu\nu}(r)$. No iteration is required.

2.5 Multicomponent EXP

The EXP approximation is a refinement of the MSA/RPA in which the total correlation functions are replaced by

$$h_{\mu\nu} \to (1 + h^{(0)}_{\mu\nu}) \exp(h_{\mu\nu} - h^{(0)}_{\mu\nu}) - 1\,, \tag{22}$$

in which $h^{(0)}_{\mu\nu}$ will be a hard sphere reference fluid in the case of MSA, and $h^{(0)}_{\mu\nu} = 0$ in the case of RPA (no hard cores). The EXP breaks unwarranted symmetries such as $h_{++} = -h_{+-}$ and ensures that $h_{\mu\nu}(r) > -1$ which is not always satisfied by the MSA/RPA. In practice EXP can be nearly as good as HNC and extends to state points where HNC doesn't have a solution.

Since the EXP approximation is defined as an action on the correlation functions in real space, a round trip through the OZ relation is required to compute the corresponding offset direct and indirect correlation functions. To do this we rewrite the OZ relation in Eq. (19) as

$$\tilde{C} = (I + \tilde{H} \cdot R)^{-1} \cdot \tilde{H}\,. \tag{23}$$

With this implemented, $c'_{\mu\nu} = c_{\mu\nu} + \beta v^{\mathrm{L}}_{\mu\nu}$ and $e'_{\mu\nu} = h_{\mu\nu} - c'_{\mu\nu}$ follow.

2.6 Structural properties

2.6.1 Pair functions

Given a converged solution pair $(c'_{\mu\nu}, e'_{\mu\nu})$, the total correlation functions can be evaluated from

$$h_{\mu\nu}(r) = c'_{\mu\nu}(r) + e'_{\mu\nu}(r) \tag{24}$$

(note that $\beta v^{\mathrm{L}}_{\mu\nu}$ cancels). The pair functions are given by $g_{\mu\nu} = \delta_{\mu\nu} + h_{\mu\nu}$.

2.6.2 Structure factors

For the partial structure factors, there is a choice of normalisation. In the present code the structure factors are perhaps unusually defined to be

$$S_{\mu\nu}(k) = \rho_\mu \delta_{\mu\nu} + \rho_\mu \rho_\nu \tilde{h}_{\mu\nu} \tag{25}$$

where $\tilde{h}_{\mu\nu} = \tilde{c}'_{\mu\nu} + \tilde{e}'_{\mu\nu}$ (again, $\beta v^{\mathrm{L}}_{\mu\nu}$ cancels); the Fourier transforms $\tilde{c}'_{\mu\nu}$ and $\tilde{e}'_{\mu\nu}$ are available as a byproduct of solving the OZ relation. With this normalisation the structure factors obey $S_{\mu\nu}(k) \to \rho_\mu \delta_{\mu\nu}$ as $k \to \infty$. An alternative (and more popular) normalisation, for which $S_{\mu\nu}(k) \to \delta_{\mu\nu}$ as $k \to \infty$, is obtained from the above expression by multiplying by $(\rho_\mu \rho_\nu)^{-1/2}$.

The unusual choice of normalisation is made to facilitate the calculation of the density-density and charge-charge structure factors *via*

$$S_{NN}(k) = \frac{\sum_{\mu\nu} S_{\mu\nu}(k)}{\sum_\mu \rho_\mu}, \quad S_{ZZ}(k) = \frac{\sum_{\mu\nu} z_\mu z_\nu S_{\mu\nu}(k)}{\sum_\mu \rho_\mu} \tag{26}$$

(assuming the ion valencies are z_μ for the latter).

2.7 Thermodynamics

The above sections specify the various closures for the integral equation problem for multicomponent systems. We provide here the suite of thermodynamic quantities which can be evaluated from the converged solution pair $(c'_{\mu\nu}, e'_{\mu\nu})$. Some care is needed in the case of hard core potentials, where $v_{\mu\nu} = \infty$ for $r < d_{\mu\nu}$.

2.7.1 Virial pressure

The so-called virial route compressibility factor is

$$\frac{\beta p}{\rho} = 1 - \frac{2\pi}{3} \sum_{\mu\nu} \rho x_\mu x_\nu \int_0^\infty dr\, r^3 \, \frac{\partial(\beta v_{\mu\nu})}{\partial r} g_{\mu\nu}(r) \tag{27}$$

(*c.f.* Eq. (1.2) in Ref. [5]). To handle the hard core contribution to this we follow the line of argument presented in Hansen and McDonald [3] and introduce the function $y_{\mu\nu} = \exp[\beta v_{\mu\nu}]\, g_{\mu\nu}$, which is continuous through into the hard core region. The above integral can be written

$$\begin{aligned} I_{\mu\nu} &= \int_0^\infty dr\, r^3\, \frac{\partial(\beta v_{\mu\nu})}{\partial r} \exp[-\beta v_{\mu\nu}]\, y_{\mu\nu}(r) \\ &= -\int_0^\infty dr\, r^3\, \frac{\partial(\exp[-\beta v_{\mu\nu}])}{\partial r}\, y_{\mu\nu}(r)\,. \end{aligned} \tag{28}$$

Let us write

$$\exp[-\beta v_{\mu\nu}] = \Theta(r - d_{\mu\nu})\, \exp[-\beta \overline{v}_{\mu\nu}] \tag{29}$$

where $\overline{v}_{\mu\nu} = v_{\mu\nu}$ for $r \geq d_{\mu\nu}$, and we don't care what it does for $r < d_{\mu\nu}$ except that it should be *continuous* through $r = d_{\mu\nu}$. The Heaviside step function $\Theta(x) = 0$ for $x < 0$ and $\Theta(x) = 1$ for $x \geq 0$, with derivative $\Theta'(x) = \delta(x)$. Then

$$\begin{aligned} I_{\mu\nu} &= -\int_0^\infty dr\, r^3 \Big(\delta(r - d_{\mu\nu}) \exp[-\beta \overline{v}_{\mu\nu}]) \\ &\qquad + \Theta(r - d_{\mu\nu}) \frac{\partial(\exp[-\beta \overline{v}_{\mu\nu}])}{\partial r} \Big)\, y_{\mu\nu}(r) \\ &= -g_{\mu\nu}(d_{\mu\nu})\, d_{\mu\nu}^3 + \int_{d_{\mu\nu}}^\infty dr\, r^3\, \frac{\partial(\beta v_{\mu\nu})}{\partial r} g_{\mu\nu}(r) \end{aligned} \tag{30}$$

In the second line we have restored $v_{\mu\nu}$ and $g_{\mu\nu}$. Thus, finally,

$$\frac{\beta p}{\rho} = 1 + \frac{2\pi}{3} \sum_{\mu\nu} \rho x_\mu x_\nu \Big[g_{\mu\nu}(d_{\mu\nu})\, d_{\mu\nu}^3 - \int_{d_{\mu\nu}}^\infty dr\, r^3\, \frac{\partial(\beta v_{\mu\nu})}{\partial r} g_{\mu\nu}(r) \Big] \tag{31}$$

This now contains explicitly the famous contact contribution to the pressure, and the integral is now restricted to the region external to the hard core. In the case of a soft potential (DPD, URPM), the contact contribution vanishes and the integral extends to $r = 0$.

In practice we write the integral contribution to the virial pressure as

$$-\frac{2\pi}{3} \int_{d_{\mu\nu}}^\infty dr\, r^3\, \frac{\partial(\sum_{\mu\nu} \rho x_\mu x_\nu\, \beta v_{\mu\nu})}{\partial r} + \sum_{\mu\nu} \rho x_\mu x_\nu t_{\mu\nu}^{\mathrm{V}} \tag{32}$$

where

$$t_{\mu\nu}^{\mathrm{V}} = -\frac{2\pi}{3} \int_{d_{\mu\nu}}^\infty dr\, r^3\, \frac{\partial(\beta v'_{\mu\nu} + \beta v_{\mu\nu}^{\mathrm{L}})}{\partial r}\, (c'_{\mu\nu} + e'_{\mu\nu}) \tag{33}$$

The first term in Eq. (32) is the result for $g_{\mu\nu} = 1$ and is the mean-field contribution. For many applications it is important to evaluate this analytically, as the individual contributions may diverge. Under the SYM condition the contribution from $v^{\rm L}_{\mu\nu}$ to the mean field term vanishes since charge neutrality implies $\sum_{\mu\nu} \rho x_\mu x_\nu v^{\rm L}_{\mu\nu} \propto v^{\rm L}(\sum_\mu \rho_\mu z_\mu)^2 = 0$. The second term in Eq. (32) is the correlation correction (note $c'_{\mu\nu} + e'_{\mu\nu} = g_{\mu\nu} - 1$).

2.7.2 Compressibility

The compressibility (not to be confused with the above *compressibility factor*) is

$$\frac{\partial(\beta p)}{\partial \rho} = 1 - \textstyle\sum_{\mu\nu} \rho x_\mu x_\nu t^{\rm C}_{\mu\nu} \tag{34}$$

where

$$t^{\rm C}_{\mu\nu} = 4\pi \int_0^\infty dr\, r^2\, (c'_{\mu\nu}(r) - \beta v^{\rm L}_{\mu\nu}) \tag{35}$$

(*c. f.* Eq. (18) in Ref. [5]). In terms of this, the isothermal compressibility is $\chi_T = [\rho(\partial p/\partial \rho)]^{-1}$. The contribution from $v^{\rm L}_{\mu\nu}$ can also often be evaluated analytically and vanishes under the SYM condition by the same argument as above. Obviously the compressibility can be integrated along an isotherm to obtain an alternative expression for the pressure. This is the so-called compressibility route to the equation of state. This expression is insensitive to the presence of hard cores since they are supposed to be contained in $v'_{\mu\nu}$ and enter *via* the direct correlation function.

2.7.3 Energy

The excess internal energy density is

$$\beta u^{\rm ex} \equiv \frac{\beta U^{\rm ex}}{V} = 2\pi\rho \textstyle\sum_{\mu\nu} \rho x_\mu x_\nu \displaystyle\int_{d_{\mu\nu}}^\infty dr\, r^2\, \beta v_{\mu\nu}(r)\, g_{\mu\nu}(r) \tag{36}$$

(*c. f.* Eq. (2.5.20) in Ref. [3]). Again this can be split into a mean field term and a correlation correction,

$$\beta u^{\rm ex} = 2\pi\rho \int_{d_{\mu\nu}}^\infty dr\, r^2\, (\textstyle\sum_{\mu\nu} \rho x_\mu x_\nu\, \beta v_{\mu\nu}) + \rho \textstyle\sum_{\mu\nu} \rho x_\mu x_\nu t^{\rm E}_{\mu\nu} \tag{37}$$

where

$$t^{\rm E}_{\mu\nu} = 2\pi \int_{d_{\mu\nu}}^\infty dr\, r^2\, (\beta v'_{\mu\nu} + \beta v^{\rm L}_{\mu\nu})\, (c'_{\mu\nu} + e'_{\mu\nu})\,. \tag{38}$$

An integration by parts shows that the mean field term here is related to that for the compressibility factor,

$$2\pi \int_{d_{\mu\nu}}^{\infty} dr\, r^2\, \beta v_{\mu\nu} = -\frac{2\pi \beta v_{\mu\nu}(d_{\mu\nu})\, d_{\mu\nu}^3}{3} - \frac{2\pi}{3} \int_{d_{\mu\nu}}^{\infty} dr\, r^3\, \frac{\partial(\beta v_{\mu\nu})}{\partial r}\,. \tag{39}$$

The equation of state can be derived from the internal energy and this is the so-called energy route.

2.7.4 Free energy and chemical potentials

Last, *only valid for the HNC closure*, are expressions for the free energy and chemical potentials. The excess free energy density can be written as

$$\begin{aligned} \beta a^{\rm ex} \equiv \frac{\beta A^{\rm ex}}{V} = \frac{1}{2} \int d^3\mathbf{r} \sum_{\mu\nu} \rho_\mu \rho_\nu (\tfrac{1}{2} h_{\mu\nu}^2 - c_{\mu\nu}) \\ + \frac{1}{2} \int \frac{d^3\mathbf{k}}{(2\pi)^3} \left[\ln\det(I - R\cdot\tilde{C}) + \mathrm{Tr}\, R\cdot\tilde{C} \right]. \end{aligned} \tag{40}$$

The first (real space) term has been simplified from the literature expressions (*c. f.* Eq. (5.1) in Ref. [6] or Eq. (30) in Ref. [7]), by inserting the HNC closure Eq. (16). The second (reciprocal space) term is couched in terms of the R and $\tilde{C}$ matrices introduced in Eq. (18): R as a diagonal matrix with entries $\rho_\mu = \rho_\mu = \rho x_\mu$ and $\tilde{C}$ is the matrix with entries $\tilde{c}_{\mu\lambda}$.

For the real space part, $\beta a^{\rm ex,1} = (\rho/2) \sum_{\mu\nu} \rho x_\mu x_\nu t_{\mu\nu}^{\rm A}$ where

$$t_{\mu\nu}^{\rm A} = 4\pi \int_0^{\infty} dr\, r^2 \left[\tfrac{1}{2} (c'_{\mu\nu}(r) + e'_{\mu\nu}(r) \right]^2 - c'_{\mu\nu}(r) + \beta v_{\mu\nu}^{\rm L}]\,. \tag{41}$$

The last term in this is same as that appearing in the compressibility (to within an overall sign) and often can be evaluated analytically.

We split the reciprocal space part into the separate contributions from the determinant and the trace. For the former we use $\ln\det M = \frac{1}{2} \sum_i \ln|\lambda_i|^2$ where the sum is over the eigenvalues λ_i of $M = I - R\cdot\tilde{C}$. Note that this matrix is not symmetric and the eigenvalues may be complex. For the latter we write the trace as $\sum_\mu \rho_\mu (\tilde{c}'_{\mu\mu} - \beta \tilde{v}_{\mu\mu}^{\rm L})$ and we note from the definition of the inverse Fourier transform in Eq. (3) that $(2\pi)^{-3} \int d^3\mathbf{k}\, \tilde{v}_{\mu\mu}^{\rm L} = v_{\mu\mu}^{\rm L}(0)$, *i. e.* the long range potential in real space evaluated at $r = 0$. Thus the reciprocal space contribution is computed from

$$\beta a^{\rm ex,2} = \frac{1}{4\pi^2} \int_0^{\infty} dk\, k^2 \left[\tfrac{1}{2} \textstyle\sum_i \ln|\lambda_i(k)|^2 + \sum_\mu \rho_\mu \tilde{c}'_{\mu\mu}(k) \right] - \tfrac{1}{2} \textstyle\sum_\mu \rho_\mu v_{\mu\mu}^{\rm L}(0)\,. \tag{42}$$

The final excess free energy density is $a^{\rm ex} = a^{\rm ex,1} + a^{\rm ex,2}$. The above expressions are insensitive to the presence of hard cores.

The corresponding chemical potentials are

$$\beta\mu_\mu^{\rm ex} = 4\pi \sum_\nu \rho x_\nu \int_0^\infty dr\, r^2 \left[\tfrac{1}{2}\, h_{\mu\nu}(r)\, e_{\mu\nu}(r) - c_{\mu\nu}(r)\right] \quad \left(\equiv \ln \gamma_\mu \right) \tag{43}$$

(*c. f.* section II C in Ref. [5]). This also defines the activity γ_μ. We write this result as $\beta\mu_\mu^{\rm ex} = \sum_\nu \rho x_\nu t_{\mu\nu}^{\rm M}$ where

$$t_{\mu\nu}^{\rm M} = 4\pi \int_0^\infty dr\, r^2 \left[\tfrac{1}{2}\, (c'_{\mu\nu} + e'_{\mu\nu})(e'_{\mu\nu} + \beta v_{\mu\nu}^{\rm L}) - c'_{\mu\nu} + \beta v_{\mu\nu}^{\rm L}\right]. \tag{44}$$

The contribution from $v_{\mu\nu}^{\rm L}$ (the last term) also appears in the free energy and is computed separately (analytically). These expressions are also insensitive to the presence of hard cores.

The virial route pressure and the chemical potentials are derived from the above HNC free energy. It therefore follows that

$$a^{\rm ex} - \textstyle\sum_\mu \rho_\mu \mu_\mu^{\rm ex} + p^{\rm ex} \equiv 0 \tag{45}$$

where $p^{\rm ex} = p - \rho$ is the excess pressure. The extent to which this actually true is largely a test of the numeric discretisation error.

3 Potentials

3.1 DPD potential

The code contains routines which specify a multicomponent potential for dissipative particle dynamics (DPD) with possible charges. The potential in this case consists of short range and long range contributions which can be mapped exactly to the Ng split. The short range part is

$$\beta v'_{\mu\nu}(r) = \begin{cases} \frac{1}{2} A_{\mu\nu}(1 - r/r_c)^2 & r < r_c \\ 0 & r \geq r_c \end{cases} . \tag{46}$$

This is a conventional choice, depending on a dimensionless symmetric repulsion amplitude matrix $A_{\mu\nu}$ and cut off beyond a distance r_c.

3.1.1 Gaussian charges

Unlike the short range part there is no consensus on the best choice for the long range interaction although the differences can always be adsorbed into a redefinition of the short range potential.[1]

The *default* choice of Gaussian charges can be supported for a range of practical reasons [1, 8]. The Gaussian charges have size $\sim \sigma$ and a density distribution $(2\pi\sigma^2)^{-3/2}\exp(-r^2/2\sigma^2)$. The corresponding interaction satisfies the SYM condition and is given by

$$v^{\mathrm{L}}_{\mu\nu}(r) = z_\mu z_\nu v^{\mathrm{L}}(r)\,, \quad \beta v^{\mathrm{L}}(r) = \frac{l_{\mathrm{B}}}{r}\,\mathrm{erf}\Big(\frac{r}{2\sigma}\Big)\,, \quad \beta v^{\mathrm{L}}(0) = \frac{l_{\mathrm{B}}}{\sigma\sqrt{\pi}}\,, \tag{47}$$

where l_{B} is the coupling strength (the Bjerrum length). The precise definition of σ is made to simplify the Fourier transform (see below).

For use with the above expressions, we need the derivative of both parts of the potential, and the Fourier transform of the long range part. These are

$$\frac{\partial(\beta v'_{\mu\nu})}{\partial r} = \begin{cases} -(A_{\mu\nu}/r_c)(1-r/r_c) & r < r_c \\ 0 & r \geq r_c \end{cases} \tag{48}$$

$$\frac{\partial(\beta v^{\mathrm{L}})}{\partial r} = -\frac{l_{\mathrm{B}}}{r^2}\,\mathrm{erf}\Big(\frac{r}{2\sigma}\Big) + \frac{l_{\mathrm{B}}}{r\sigma\sqrt{\pi}}\,e^{-r^2/4\sigma^2}\,, \tag{49}$$

and

$$\beta\tilde{v}^{\mathrm{L}}(k) = \frac{4\pi l_{\mathrm{B}}}{k^2} \times e^{-k^2\sigma^2}\,. \tag{50}$$

The mean field contributions to the virial route pressure and energy are given by (see also Eq. (39))

$$2\pi\int_0^\infty dr\, r^2\,(\textstyle\sum_{\mu\nu}\rho x_\mu x_\nu\,\beta v_{\mu\nu}) = \frac{\pi r_c^3}{30}\textstyle\sum_{\mu\nu}\rho x_\mu x_\nu A_{\mu\nu}\,. \tag{51}$$

Because of the SYM condition, the contribution from v^{L} vanishes from this, and also vanishes from the mean field contributions to the compressibility and chemical potentials (this is true for all the cases in this section).

This DPD model depends on the dimensionless repulsion amplitude matrix $A_{\mu\nu}$, the ratios l_{B}/σ and σ/r_c, and the dimensionless density ρr_c^3 [1]. In these terms $r_c = 1$ is often used as the fundamental length scale. The ratio l_{B}/σ plays the role of an inverse effective temperature for the Coloumbic part of the potential.

[1] By writing $\beta\tilde{v}^{\mathrm{L}}(k) = 4\pi l_{\mathrm{B}}k^{-2}(1+k^2\sigma^2/n)^{-n}$, there is a natural hierarchy going from Bessel ($n=1$), approximate ($n=2$) and exact ($n=4$) Slater, Gaussian ($n\to\infty$).

3.1.2 Bessel charges

If Bessel charges are selected the charge distribution becomes $K_1(r/\sigma)/2\pi^2\sigma^2 r$ and the expressions change to [8]

$$\beta v^{\mathrm{L}}(r) = \frac{l_{\mathrm{B}}}{r}(1 - e^{-r/\sigma})\,, \quad \beta v^{\mathrm{L}}(0) = \frac{l_{\mathrm{B}}}{\sigma}\,, \tag{52}$$

$$\frac{\partial(\beta v^{\mathrm{L}})}{\partial r} = -\frac{l_{\mathrm{B}}}{r^2}\Big[1 - e^{-r/\sigma}\Big(1 + \frac{r}{\sigma}\Big)\Big]\,, \tag{53}$$

$$\beta\tilde{v}^{\mathrm{L}}(k) = \frac{4\pi l_{\mathrm{B}}}{k^2} \times \frac{1}{1 + k^2\sigma^2}\,. \tag{54}$$

Note that σ here is chosen to match the second moment of the charge distribution for the Gaussian case.

3.1.3 Linear charge smearing

Another alternative is linear charge smearing proposed by Groot [9]. In this case the charge distribution is $(3/\pi R^3)(1 - r/R)$ for $r < R$, vanishing for $r > R$. This gives rise to an interaction potential in reciprocal space [8]

$$\beta\tilde{v}^{\mathrm{L}}(k) = \frac{4\pi l_{\mathrm{B}}}{k^2}\Big(\frac{24 - 24\cos kR - 12kR\sin kR}{k^4R^4}\Big)^2\,. \tag{55}$$

A closed form expression in real space is not available, hence is not implemented in the code. This means that the thermodynamics is *not available* in this case, though the HNC solution goes through since this only requires the reciprocal space potential. The term in large brackets is the charge density in reciprocal space. Matching the second moment of the charge distribution [8] shows that the equivalent Gaussian smearing length is $\sigma = R\sqrt{2/15}$.

3.1.4 Approximate Slater smearing

Another popular alternative is Slater smearing proposed by González-Melchor *et al.* [10]. They use an approximate expression for the interaction potential, for which we have

$$\beta v^{\mathrm{L}}(r) = \frac{l_{\mathrm{B}}}{r}\Big[1 - e^{-2ßr}(1 + ßr)\Big]\,, \quad \beta v^{\mathrm{L}}(0) = ßl_{\mathrm{B}}\,, \tag{56}$$

$$\frac{\partial(\beta v^{\mathrm{L}})}{\partial r} = -\frac{l_{\mathrm{B}}}{r^2}\Big[1 - e^{-2ßr}(1 + 2ßr(1 + ßr))\Big]\,, \tag{57}$$

$$\beta\tilde{v}^{\mathrm{L}}(k) = \frac{4\pi l_{\mathrm{B}}}{k^2} \times \frac{1}{(1 + k^2/4ß^2)^2}\,. \tag{58}$$

These actually corresponds to a charge density $ß^2 e^{-2ßr}/\pi r$ where ß is a smearing parameter, not to be confused with $\beta = 1/k_{\mathrm{B}}T$. The equivalent Gaussian smearing length is here $\sigma = 1/ß\sqrt{2}$.

3.1.5 Exact Slater smearing

The intended charge density in Slater smearing is $(1/\pi\lambda^3)e^{-2r/\lambda}$. It turns out that there are exact expressions for the interaction between such charges [8],

$$\beta v^{\mathrm{L}}(r) = \frac{l_{\mathrm{B}}}{r}\Big[1 - e^{-2r/\lambda}\Big(1 + \frac{11r}{8\lambda} + \frac{3r^2}{4\lambda^2} + \frac{r^3}{6\lambda^3}\Big)\Big], \quad \beta v^{\mathrm{L}}(0) = \frac{5l_{\mathrm{B}}}{8\lambda}, \tag{59}$$

$$\frac{\partial(\beta v^{\mathrm{L}})}{\partial r} = -\frac{l_{\mathrm{B}}}{r^2}\Big[1 - e^{-2r/\lambda}\Big(1 + \frac{2r}{\lambda} + \frac{2r^2}{\lambda^2} + \frac{7r^3}{6\lambda^3} + \frac{r^4}{3\lambda^4}\Big)\Big], \tag{60}$$

$$\beta\tilde{v}^{\mathrm{L}}(k) = \frac{4\pi l_{\mathrm{B}}}{k^2} \times \frac{1}{(1 + k^2\lambda^2/4)^4}. \tag{61}$$

The equivalent Gaussian smearing length for these exact expressions is $\sigma = \lambda$. Originally, González-Melchor *et al.* [10] set $ß = 1/\lambda$ in Eq. (56), but setting $ß = 5/(8\lambda)$ matches the values of $\beta v^{\mathrm{L}}(0)$.

3.2 URPM and softened URPM potential

Another potential provided by the code at present is for the URPM in which the unlike pair potential is artificially softened. The standard URPM (an equimolar mixture of Gaussian charges) is given by the DPD potential above with $z_\mu = \pm 1$ and $A_{\mu\nu} = 0$. There are two ways the softened version can be implemented. The first way is to work purely in reciprocal space. It is important to note that this takes the model away from the SYM condition expressed in Eq. (15). In this approach the short range part $v'_{\mu\nu} = 0$, and the long range part is

$$\beta v^{\mathrm{L}}_{11} = v^{\mathrm{L}}_{22} = \frac{l_{\mathrm{B}}}{r}\,\mathrm{erf}\Big(\frac{r}{2\sigma}\Big) \quad \beta v^{\mathrm{L}}_{12} = -\frac{l_{\mathrm{B}}}{r}\,\mathrm{erf}\Big(\frac{r}{2\sigma'}\Big) \tag{62}$$

where typically $\sigma' > \sigma$ (in the case $\sigma' = \sigma$ we have the original URPM which is also contained in the DPD potential described above). The value at $r = 0$, the potential in reciprocal space, and the derivatives follow *mutatis mutandis* from the DPD case.

In the alternative approach we retain the SYM condition so that the long range part is given by Eq. (47) and the short range part is

$$\beta v'_{12} \equiv -\beta\Delta v_{12} = -\frac{l_{\mathrm{B}}}{r}\,\mathrm{erf}\Big(\frac{r}{2\sigma'}\Big) + \frac{l_{\mathrm{B}}k_{\mathrm{B}}T}{r}\,\mathrm{erf}\Big(\frac{r}{2\sigma}\Big) \tag{63}$$

(obviously, $v'_{11} = v'_{22} = 0$). We define Δv_{12} as the potential that should be *added* to the softened URPM, to recover the standard URPM. This is so that we can use the softened version as a reference fluid.

For both routes the mean field contributions to the virial route pressure and energy are non-zero only for the unlike pairs. We can evaluate them with v'_{12} given by Eq. (63)

$$-\frac{2\pi}{3}\int_0^\infty dr\, r^3\, \frac{\partial(\beta v'_{12})}{\partial r} = 2\pi \int_0^\infty dr\, r^2\, \beta v'_{12} = 2\pi l_{\mathrm{B}}(\sigma'^2 - \sigma^2)\,. \qquad (64)$$

If we take the first, purely reciprocal space route, the failure to satisfy the SYM condition means the compressibility and chemical potentials need to take account of $v^{\mathrm{L}}_{\mu\nu}$. This leads to the long range contributions

$$4\pi \int_0^\infty dr\, r^2\, \beta v'_{12} = 4\pi l_{\mathrm{B}}(\sigma'^2 - \sigma^2)\,. \qquad (65)$$

These should not be included if we follow the second route.

3.3 RPM and softened RPM potential

Another potential which can be solved in the code is for charged hard spheres: both the 'vanilla' RPM, and a version in which the unlike pair potential is artificially softened, are provided. For charged hard spheres the short range interaction potential can be taken to be defined by $e^{-\beta v'} = 0$ for $r \le \sigma$ and $e^{-\beta v'} = 1$ for $r > \sigma$, where σ is the hard core diameter.

Given this, we might naïvely think of using l_{B}/r as the long range part of the potential. Although the divergence at $r \to 0$ is hidden inside the hard core, this choice forces the offset functions $c'_{\mu\nu}$ and $e'_{\mu\nu}$ to try to compensate, giving rise to numerical discretisation artefacts (note that the actual direct and indirect correlation functions, $c_{\mu\nu}$ and $e_{\mu\nu}$, are non-vanishing and finite inside the hard cores). Numerically much better is to chose the long range potential such that it is bounded inside an inner hard core region. By far the simplest choice is a truncated Coulomb law,

$$\beta v^{\mathrm{L}}(r) = \begin{cases} l_{\mathrm{B}}/\sigma & r \le \sigma\,, \\ l_{\mathrm{B}}/r & r > \sigma\,, \end{cases} \qquad \beta v^{\mathrm{L}}(0) = \frac{l_{\mathrm{B}}}{\sigma}\,. \qquad (66)$$

This does least damage to the potential outside the hard cores so we can continue to use our simple definition of $e^{-\beta v'}$ above. With this choice, the Fourier transform is easily shown to be

$$\beta \tilde{v}^{\mathrm{L}} = \frac{4\pi l_{\mathrm{B}}}{k^2} \times \frac{\sin k\sigma}{k\sigma}\,. \qquad (67)$$

The RPM potential in the code is implemented using these choices. However this choice has a knock-on effect on the thermodynamics discussed below.

As in the softened URPM, there are two ways the softened RPM can be implemented. The first way is to work purely in reciprocal space which takes the model away from the SYM condition expressed in Eq. (15). In this approach we change the long range part between unlike charges to

$$\beta v_{12}^{\mathrm{L}} = -\frac{l_{\mathrm{B}}}{r}\,\mathrm{erf}(\kappa r)\,. \tag{68}$$

The potential in reciprocal space and the derivatives follow *mutatis mutandis* from the softened URPM case (*i. e.* URPM $\sigma \to 0$ and $\sigma' \to 1/(2\kappa)$). Unlike the naïve l_{B}/r choice for the vanilla RPM, this softened potential is well-behaved inside the hard core and can be used without further modification.

In the alternative approach we retain the SYM condition so that the long range part is given as in the vanilla RPM above, and the short range part is

$$\beta v'_{12} \equiv -\beta\Delta v_{12} = \frac{l_{\mathrm{B}}}{r}\,\mathrm{erfc}(\kappa r) \quad (r > \sigma)\,, \tag{69}$$

together with $\exp[-\beta v'_{12}] = 0$ for $r \leq \sigma$. As in the URPM, we define Δv_{12} as the potential that should be *added* to the softened RPM, to recover the standard RPM.

For both routes the mean field contributions to the virial route pressure and energy are non-zero only for the unlike pairs. We can evaluate them with v'_{12} given by Eq. (63)

$$-\frac{2\pi}{3}\int_\sigma^\infty dr\, r^3\,\frac{\partial(\beta v'_{12})}{\partial r} = \pi l_{\mathrm{B}}\Big(\frac{\sigma\exp[-\kappa^2\sigma^2]}{\kappa\sqrt{\pi}} + \Big(\frac{1}{2\kappa^2} - \frac{\sigma^2}{3}\Big)\,\mathrm{erfc}(\kappa\sigma)\Big)\,. \tag{70}$$

$$2\pi\int_\sigma^\infty dr\, r^3\,\beta v'_{12} = \pi l_{\mathrm{B}}\Big(\frac{\sigma\exp[-\kappa^2\sigma^2]}{\kappa\sqrt{\pi}} + \Big(\frac{1}{2\kappa^2} - \sigma^2\Big)\,\mathrm{erfc}(\kappa\sigma)\Big)\,. \tag{71}$$

In the limit $\sigma \to 0$ these both become $\pi l_{\mathrm{B}}/2\kappa^2$ (*c. f.* softened URPM). In the limit $\kappa \to \infty$ (RPM case) both vanish.

If we take the first, purely reciprocal space route, the failure to satisfy the SYM condition means the compressibility and chemical potentials need to take account of $v_{\mu\nu}^{\mathrm{L}}$. This leads to a contribution for unlike pairs

$$4\pi\int_0^\infty dr\, r^2\,\Big[\frac{l_{\mathrm{B}}}{r}\,\mathrm{erfc}(\kappa r) + \Big(\frac{l_{\mathrm{B}}}{\sigma} - \frac{l_{\mathrm{B}}}{r}\Big)\Theta(\sigma - r)\Big] = \frac{\pi l_{\mathrm{B}}}{\kappa^2} - \frac{2\pi l_{\mathrm{B}}\sigma^2}{3}\,. \tag{72}$$

The second term accounts for the choice of v^{L} in the core.

4 Implementation notes

Most of the code is self-expanatory, given the above mathematical background. Correlation functions in the real space domain are discretised in an array of size $i = 1 \ldots n-1$ with a spacing δ_r so that $r_i = \delta_r \times i$. Usually one choses δ_r to be a small fraction of the relevant length scale (*e. g.* $\delta_r = 0.01\, r_c$) and $n\delta_r$ to be some large multiple of the same (*e. g.* $n\delta_r \approx 40\, r_c$). The actual value of n can be chosen to optimise the fast Fourier transform algorithm, for instance $n = 4096$. The concomitant functions in the reciprocal space domain are discretised in an array of size $j = 1 \ldots n-1$ with a spacing $\delta_k \equiv \pi/(n\delta_r)$ so that $k_j = \delta_k \times j$. The array size $n-1$ and the value δ_k are fixed by the demands of the fast Fourier transform algorithm.

The implementation of the Fourier transforms is as follows. A discrete version of Eq. (2) is

$$\tilde{h}_j = (2\pi\delta_r/k_j) \times 2 \textstyle\sum_{i=1}^{n-1} r_i h_i \sin(\pi ij/n) \tag{73}$$

where $j = 1 \ldots n-1$. The quantity $2 \sum_{i=1}^{n-1} r_i h_i \sin(\pi ij/n)$ is computed by calling the FFTW routine `RODFT00` on $r_i h_i$. From the FFTW documenation, this specific routine works best when the array length is of the form $2^a 3^b 5^c 7^d 11^e 13^f - 1$ where $e + f$ is either 0 or 1 and the other exponents are arbitrary (this means n should be of the form of the first term since the array length is $n-1$). For the back-transform the corresponding discrete version of Eq. (4) is

$$h_i = (\delta_k/(2\pi)^2 r_i) \times 2 \textstyle\sum_{j=1}^{n-1} k_j \tilde{h}_j \sin(\pi ij/n) \tag{74}$$

The quantity $2 \sum_{j=1}^{n-1} k_j h_j \sin(\pi ij/n)$ is computed by calling `RODFT00` on $k_j h_j$.

Versions ≥ 1.7 of the code can handle an arbitrary number of components. A species pair array index (i, j) is mapped to a 'function' index s according to the following rule

$$s = \begin{cases} i + j(j-1)/2 & \text{if } i \leq j\,, \\ j + i(i-1)/2 & \text{if } i > j\,. \end{cases} \tag{75}$$

This is implemented in various forms throughout the code. The mapping essentially labels the entries in the upper triangular form of a symmetric matrix as shown here (where i labels rows, and j labels columns)

	1	2	3	4	$\cdots$
1	1	2	4	7	$\cdots$
2		3	5	8	$\cdots$
3			6	9	$\cdots$
4				10	$\cdots$
$\cdots$					$\cdots$

(76)

If necessary, one can recover the array indices (i, j) from the the index s by the following rule, where $\lfloor \cdots \rfloor$ is the 'floor' function,

$$j = \left\lfloor \frac{1 + \sqrt{8s - 7}}{2} \right\rfloor , \quad i = n - j(j-1)/2 \, . \tag{77}$$

This is not used in the code though. The proof is left as an exercise.

The matrix multiplications in the Ornstein-Zernike relation and elsewhere are implemented using the standard `MATMUL` function in FORTRAN 90. The matrix inversions are not implemented *per se*, but are rather done by solving the matrix equation $A \cdot X = B$ using a bespoke subroutine `axeqb_reduce` in the code which implements a simplified version of Gauss-Jordan elimination with pivoting (but not LDU factorisation) [11]. The eigenvalues of the (non-symmetric) matrix required for the HNC free energy are found by calling `DGEEV` in the LAPACK library.

The Ng acceleration scheme is implemented by keeping track of a number of previous iterations (typically 6), recycling the storage. Thus the main functions $c'_{\mu\nu}$ and $e'_{\mu\nu}$ are multidimensional: the first axis is the spatial discretisation, the second axis is the species pair index, and the third axis is the Ng trajectory index (cyclical history array). The Ng scheme involves also involves solving $A \cdot X = B$ but this is done with a call to `DSYSV` in the LAPACK library (which does implement LDU factorisation). Now you might think you could use `DSYSV` for the matrix operations in the OZ inversions, or `axeqb_reduce` in the Ng accelerator, but for some reason connected to the LDU factorisation this leads to numerical inaccuracies. If anyone can shed any light on this please let me know!

The thermodynamic quantities in section 2.7 are all evaluated by standard trapezium rules, applied to the integrals given therein.

If scripting with `python` note that the `python` array index is one less than the FORTRAN array index. This conversion is seamlessly and invisibly implemented in `f2py`. Also note that the optional parameters in the FORTRAN function calls can be omitted also in `python`, and the code will gracefully fall back to the documented defaults.

5 Usage notes

The code is presented as a suite of functions in a FORTRAN 90 module. Thus it typically needs a driver code, which can be written in FORTRAN 90 or in `python` with the use of `f2py` (see examples below). Thermodynamic integration schemes, for example the compressibility route to the equation of state, are not implemented. Rather the philosophy has been to focus on robustly solving the basic integral equation problem.

5.1 Routines

- `initialise` – Allocate all arrays given the number of components `ncomp`, the number of real space points `ng`, and the length of the Ng trajectory `nps`. Initialise the FFTW plan for fast Fourier transforms. This `initialise` routine is typically called once, at the start, after the values of `ncomp`, `ng`, `nps`, and `deltar` have been fixed.

- `dpd_potential(charge_type)` – Sets up the DPD potential described in section 3.1. This routine can be called multiple times. The optional integer parameter indicates whether the potential should be initialised with Gaussian charges (`1`), Bessel charges (`2`), linear (Groot) charges (`3`), Slater charges with approximate interaction (`4`), Slater charges with exact interaction (`5`). The linear case (`3`) is currently incomplete in the sense that the thermodynamics is not calculated. If omitted, the `charge_type` defaults to Gaussian.

- `urpm_potential(use_ushort)` – Sets up the ultrasoft restricted primitive model (URPM) potential with optional softening as described in section 3.2. This routine can be called multiple times. It checks that `ncomp = 2` and forces $z_1 = 1$ and $z_2 = -1$. The optional logical parameter `use_ushort` (default `.false.`) indicates whether the potential should be formulated to contain a short range real space correction.

- `rpm_potential(use_ushort)` – Sets up the restricted primitive model (RPM) with optional softening as described in section 3.3. This routine can be called multiple times. It checks that `ncomp = 2` or `ncomp = 3` and forces $z_1 = 1$, $z_2 = -1$, and $z_3 = 0$ as required (so the third component is neutral hard spheres). The optional logical parameter works the same way as in the URPM potential. The hard core diameters are either taken from the array `diam(:, :)`, or if these are all zero they are set to a common value `sigma`. The parameter `sigma` itself is used for the *inner* core diameter which features in the definition of v^{L} in Eq. (66). Thus, by setting `sigma` and separately `diam(:, :)`, one can implement models with dissimilar hard sphere radii, or even non-additive hard sphere models. For convenience if `sigma` was zero (*i. e.* the initial default) then it is re-set to the minimum of `diam(:, :)`.

- `hs_potential` – Sets up hard sphere potentials for an arbitrary number of components, with common diameter `sigma`. This routine can be called multiple times.

- `hnc_solve` – Calculate the HNC solution. Specifically, initialise $c'_{\mu\nu}$ if the logical flag `cold_start` is set, take enough Picard steps to pump-prime the Ng acceleration method, then attempt to converge to a solution. This is the basic HNC solver routine and contains appropriate calls to `oz_solve`, `hnc_picard`, `hnc_ng` and `conv_test` below. A warning is issued if the convergence criterion is not met. If the logical flag `auto_fns` is also set (the default) the three `make_*` functions are also called, resulting in a complete solution to the problem. The function `hnc_solve` should be called only after `initialise` and the potential has been initialised. It may be called multiple times with different densities for a given potential, or the potential can be reset between calls. In later calls, the existing $c'_{\mu\nu}$ is used as a starting point unless the logical flag `cold_start` is reset.

- `msa_solve` – Calculate the MSA solution, using a similar iterative algorithm to the HNC closure. This routine is the basic MSA solver and contains appropriate calls to `oz_solve`, `msa_picard`, `msa_ng` and `conv_test`. A warning is issued if the convergence criterion is not met. If the logical flag `auto_fns` is set (the default) the three `make_*` functions are also called, resulting in a complete solution to the problem. Like the HNC, the function `msa_solve` should be called only after `initialise` and the potential has been initialised. It may be called multiple times with different densities for a given potential, or the potential can be reset between calls. In these subsequent calls, the existing $c'_{\mu\nu}$ within the hard core regions is used as a starting point unless the logical flag `cold_start` is reset. In all cases though, $c'_{\mu\nu}$ outwith the hard core regions is set equal to $-\beta v'_{\mu\nu}$ for the current potential. Note that the MSA is the same as the RPA if there are no hard cores (therefore, use `rpa_solve` in that case!).

- `rpa_solve` – Calculate the RPA solution. This involves only a single round trip through `oz_solve` from $c'_{\mu\nu} = -\beta v'_{\mu\nu}$. Provided the logical flag `auto_fns` is set (the default), the three `make_*` functions are also called. The function `rpa_solve` should be called only after `initialise` and the potential has been initialised. It may be called multiple times with different densities for a given potential, or the potential can be reset between calls. The RPA closure cannot be used with hard cores (see section 2.4).

- `save_reference` – In the presence of hard cores, EXP requires a reference state $h^{(0)}_{\mu\nu}$, and this routine saves the current $h_{\mu\nu}$ solution. Usually, this reference state will be the MSA solution to hard sphere problem

without the tail potential, however this is not enforced. See next entry for a note on the usage.

- `exp_refine` – Implement the EXP refinement. Use Eq. (22) to compute a refined $h_{\mu\nu}$ from the current solution and the reference $h^{(0)}_{\mu\nu}$, then call `oz_solve2` to rebuild the complete solution according to Eq. (23). Provided the logical flag `auto_fns` is set (the default), the three `make_*` functions are also called. A call to `initialise` sets $h^{(0)}_{\mu\nu} = 0$, thus `exp_refine` is suitable for immediate use with soft potentials without hard cores. To use `exp_refine` for potentials with hard cores, calculate and save the reference state, for example in FORTRAN 90:

```
[ .. remove tail, eg lb = 0.0 or call hs_potential ]
call msa_solve
call save_reference
[ .. restore tail, eg lb > 0.0 ]
call msa_solve
call exp_refine
```

  Note that `exp_refine` restores the reference state after completing so it can be called multiple times with the same reference state.

- `oz_solve` – Solve the OZ relation in the form of Eq. (20) to find $e'_{\mu\nu}$ from $c'_{\mu\nu}$. As a byproduct, this generates $\tilde{e}'_{\mu\nu}$ and $\tilde{c}'_{\mu\nu}$.

- `oz_solve2` – Solve the OZ relation in the form of Eq. (23) to find $e'_{\mu\nu}$ and $c'_{\mu\nu}$ from the *reference* state $h^{(0)}_{\mu\nu}$. As a byproduct generate $\tilde{e}'_{\mu\nu}$ and $\tilde{c}'_{\mu\nu}$. Store the result in the first position in the cyclical history array.

- `hnc_picard` – Solve the HNC closure to find $c'_{\mu\nu}$ from $e'_{\mu\nu}$, and take a Picard step in the sense of mixing a fraction `alpha` of the new $c'_{\mu\nu}$ with the old $c'_{\mu\nu}$.

- `hnc_ng` – Solve the HNC closure to find a new $c'_{\mu\nu}$ given $e'_{\mu\nu}$, but using the Ng acceleration scheme.

- `msa_picard` – Solve the MSA closure to find $c'_{\mu\nu}$ from $e'_{\mu\nu}$, and take a Picard step in the sense of mixing a fraction `alpha` of the new $c'_{\mu\nu}$ with the old $c'_{\mu\nu}$.

- `msa_ng` – Solve the MSA closure to find a new $c'_{\mu\nu}$ given $e'_{\mu\nu}$, but using the Ng acceleration scheme.

- `conv_test` – Check convergence in iterative schemes by calculating the difference (saved in `error`) between the current and immediately preceding $c'_{\mu\nu}$.
- `make_pair_functions` – from Eq. (24). The choice to present $h_{\mu\nu}$ rather $g_{\mu\nu}$ is made because there may be interest in the asymptotic behaviour of $h_{\mu\nu} \to 0$ as $r \to \infty$, and it is assumed the user can add '1' to get the pair distribution functions.
- `make_structure_factors` – from Eq. (25).
- `make_thermodynamics` – everything in section 2.7.
- `hnc_kspace_integrals` – specialised calculation of reciprocal space integrals in section 2.7.4 (called by `make_thermodynamics`).
- `write_params` – Write out basic information. Often useful as a check that the right potential is being solved.
- `write_thermodynamics` – Write out all thermodynamic quantities, assuming `make_thermodynamics` has been called.

Note that the three `make_*` routines are independent and can be called in any order.

5.2 User parameters

- `ncomp` – The number of component species, arbitrary ≥ 1 (default 1).
- `ng` – The real space grid size, n, usually a power of 2 (default 4096).
- `nps` – The length of the Ng trajectory, equivalently the size of the cyclical history array (usually safe to leave at the default 6).
- `npic` – The number of Picard steps to pump-prime the Ng acceleration scheme (usually safe to leave at the default 6, but should be $\geq$ `nps`).
- `maxsteps` – maximum number of steps to take before giving up on convergence (usually safe to leave at default 100).
- `verbose` – Logical flag to print solver diagnostics (default `.false.`).
- `silent` – Suppress printing of warning/error messages (default `.true.`).

- `cold_start` – Logical flag to indicate whether the main solvers should execute a cold start by initialising $c'_{\mu\nu}$. This flag is initially set `.true.`, then set `.false.` after a main solver call since it is often the case that a subsequent problem can re-use the current solution as a good initial guess. The flag can of course be reset `.true.` by the user at any point.

- `start_type` – In a HNC cold start initialise $c'_{\mu\nu} = 0$ (start type 1), $c'_{\mu\nu} = -\beta v'_{\mu\nu}$ (start type 2), or $c'_{\mu\nu} = \exp(-\beta v'_{\mu\nu}) - 1$ (start type 3, default). Usually safe to leave at default.

- `auto_fns` – Logical flag (default `.true.`) whether to call the `make_*` routines on exit of the main solver routines. Since these routines are relatively inexpensive one would usually leave this flag set.

- `deltar` – The real space grid spacing δ_r (default 0.01).

- `alpha` – Fraction of new solution in Picard method (usually safe to leave at default 0.2).

- `tol` – Error tolerance for claiming convergence (usually safe to leave at default 10^{-12}).

- `rc` – DPD repulsion range r_c (default 1.0).

- `lb` – Coulomb coupling strength l_B (default 0.0)

- `sigma` – smearing length σ (default 1.0) in the case of Gaussian or Bessel charges in the DPD potential and URPM potentials; default or inner core diameter σ in the case of the RPM, or hard sphere potentials.

- `kappa` – parameter κ (default -1) in the softened RPM potential. A negative value is interpreted as $\kappa \to \infty$ and generates the vanilla RPM.

- `sigmap` – charge size σ' (default 1.0), used for the softened URPM potential.

- `rgroot` – linear (Groot) charge smearing range R (default 1.0) in the DPD potential (`charge_type = 3`).

- `beta` – Slater charge smearing parameter ß (default 1.0) in the DPD potential with approximate interaction (`charge_type = 4`). One would usually calculate this as ß $= 1/\lambda$, or (better) ß $= 5/(8\lambda)$.

- `lbda` – Slater charge smearing length λ (default 1.0) in the DPD potential with exact interaction (`charge_type = 5`). The name is truncated because `lambda` is a reserved word in python.

- `rho(:)` – An array of size `ncomp` where `rho(mu)` is the density ρ_μ. All entries default to zero after a call to `initialise`.

- `z(:)` – An array of size `ncomp` where `z(mu)` is the valence z_μ. All entries default to zero after a call to `initialise`.

- `diam(:, :)` – An array of size `ncomp`×`ncomp` where `diam(mu, nu)` is the hard core core diameter $\sigma_{\mu\nu}$ (currently used for RPM and HS potentials). Only elements above the diagonal are used, *e. g.* `diam(1,2)`, `diam(1,3)`, and `diam(2,3)`. Elements below the diagonal are ignored. All entries default to zero after a call to `initialise`.

- `arep(:, :)` – An array of size `ncomp`×`ncomp` where `arep(mu, nu)` is the pairwise repulsion amplitude $A_{\mu\nu}$ used in the DPD potential. Only elements above the diagonal are used, *e. g.* `arep(1,2)`, `arep(1,3)`, and `arep(2,3)`. Elements below the diagonal are ignored. All entries default to zero after a call to `initialise`.

5.3 Outputs

- `deltak` – The reciprocal space grid spacing $\delta_k = \pi/(n\delta_k)$, fixed by the needs of the fast Fourier transform algorithm. Available after a call to `initialise`.

- `r(:)` – The array `r(i)` contains $r_i = i \times \delta_r$ for $i = 1 \ldots n-1$. Available after a call to `initialise`.

- `k(:)` – The array `k(j)` contains $k_j = j \times \delta_k$ for $j = 1 \ldots n-1$. Available after a call to `initialise`.

- `hr(:, :, :)` – An array containing the total correlation functions $h_{\mu\nu}(r)$ as in Eq. (24). Specifically `hr(i, mu, nu)` contains $h_{\mu\nu}(r_i)$ where $r_i = i \times \delta_r$ for $i = 1 \ldots n-1$. The values at $r = 0$ can be obtained by linear extrapolation [4]. These arrays are available after a call to `make_pair_functions` which is done automatically by the solver routines unless `auto_fns` is turned off.

- `sk(:, :, :)` – An array containing structure factors $S_{\mu\nu}(k)$ as in Eq. (25). Specifically `sk(j, mu, nu)` contains $S_{\mu\nu}(k_j)$ where $k_j = j \times \delta_k$ for $j = 1 \ldots n-1$. These arrays are available after a call to `make_structure_functions` which is done automatically by the solver routines unless `auto_fns` is turned off.

- `error` – The final convergence error estimate for the iterative solvers, should be less than `tol` if converged.

- `return_code` – Integer indicates success if `return_code = 0`, otherwise an error occurred (see next).

- `error_msg` – String (character array) describing the problem in the case that `return_code` is non-zero. The message is printed out anyway unless the `silent` flag is set. Note that in `python` a returned string appears as fixed length binary array and needs massaging to get into a usable form, thus for example:

```
#!/usr/bin/env python3
from oz import wizard as w
...
if w.return_code:
    msg = str(w.error_msg, 'utf-8').strip()
    raise Exception(msg)
```

- `closure_name` , `model_name` , `version` – Strings (character arrays) giving respectively the last-used closure (a three-letter acronym), the potential model last used to set the potential arrays, and the SunlightHNC version number (added in 1.10).

The following quantities are available after a call to `make_thermodynamics` (which is done automatically unless `auto_fns` is turned off). They are also reported by `write_thermodynamics`.

- `press` , `pex` – The virial route pressure βp, and excess pressure $\beta p^{\text{ex}} = \beta p - \beta\rho$

- `cf_mf` , `cf_gc` , `cf_xc` – The mean-field, contact, and correlation contributions respectively, to the virial route compressibility factor $\beta p/\rho$ (thus βp^{ex} is the sum of these multiplied by ρ).

- `comp` , `comp_xc` – The compressibility $\partial(\beta p)/\partial\rho$ (not to be confused with the above), and the correlation contribution to the same.

- `uex` , `uex_mf` , `uex_corrn` – Excess internal energy density βu^{ex} (*i. e.* excluding kinetic contribution); mean field and correlation terms.

The following are calculated only for the HNC closure.

- `aex_rs`, `aex_rl` – The short and long range (v^{L} term) pieces of the real space contribution to the excess free energy density.
- `aex_kd`, `aex_ks`, `aex_kl` – The determinant, and trace broken into short and long range (v^{L} term) pieces, for the reciprocal space contribution to the excess free energy density.
- `aex` – The total excess free energy density βa^{ex} (sum of all the above).
- `muex(:)` – Chemical potentials, `muex(mu)` contains $\beta\mu_\mu^{\mathrm{ex}}$
- `deficit` – The quantity $a^{\mathrm{ex}} - \sum_\mu \rho_\mu \mu_\mu^{\mathrm{ex}} + p^{\mathrm{ex}}$ (which should vanish). This can be viewed as a certificate of the numerical discretisation error.

The following internal variables may be useful.

- `c(:, :, 1)`, `e(:, :, 1)` – After a successful call to one of the main solver routines, these contain the most converged solution, such that for example `c(i, s, 1)` contains $c'_{\mu\nu}(r_i)$ $(= c_{\mu\nu} + \beta v^{\mathrm{L}}_{\mu\nu})$, where `s` is the species pair index as in Eq. (75).
- `ck(:, :)`, `ek(:, :)` – Arrays containing the Fourier-transformed offset direct and indirect correlation functions, such that for example `ck(j, s)` contains $\tilde{c}'_{\mu\nu}(k_j)$ where `s` is the species pair index.
- `h0(:, :)` – An array containing the reference state total correlation function, such that `h0(i, s)` contains $h^{(0)}_{\mu\nu}(r_i)$ where `s` is the species pair index. Used by `exp_refine` and `oz_solve2`.
- `ushort(:, :)`, `ulong(:, :)` – Arrays containing the potential, such that `ushort(i, s)` contains $\beta v'_{\mu\nu}(r_i)$ and `ulong(i, s)` contains $\beta v^{\mathrm{L}}_{\mu\nu}(r_i)$, where `s` is the species index.
- `expnegus(:, :)` – A similar array containing $\exp(-\beta v'_{\mu\nu})$.
- `dushort(:, :)`, `dulong(:, :)` – Similar arrays containing the derivatives of the potential, $\partial(\beta v'_{\mu\nu})/\partial r$ and $\partial(\beta v^{\mathrm{L}}_{\mu\nu})/\partial r$.
- `ulongk(:, :)` – Array containing the Fourier transform of the long range part of the potential, such that `ulongk(j, s)` contains $\beta \tilde{v}^{\mathrm{L}}_{\mu\nu}(k_j)$, where `s` is the species index.
- `u0(:)` – Long range potential at overlap, `u0(mu)` contains $\beta v^{\mathrm{L}}_{\mu\mu}(0)$.
- `pi`, `twopi`, `fourpi` — Available for convenience $(= \pi, 2\pi, 4\pi)$.

Of these, the potential parts may of course be initialised in the driver routine, as may the reference state total correlation function in which latter case `oz_solve2` acts as a general OZ solver. Note that the HNC, MSA and RPA solvers require *only* the long range potential in reciprocal space, $\beta\tilde{v}^{\mathrm{L}}_{\mu\nu}$ (`ulongk`), and the exponentiated short range potential, $\exp(-\beta v'_{\mu\nu})$ (`expnegus`), thus a very minimalist approach can be taken to specifying the potential. The EXP solver uses in addition the long range potential in real space, $\beta v^{\mathrm{L}}_{\mu\nu}$ (`ulong`). The full suite of thermodynamics calculations do require all the remaining potential functions though.

6 Examples

6.1 Virial route pressure

As a basic example the following minimalist code solves the HNC closure problem for standard one-component DPD at $\rho = 3$ and $A = 25$, and prints the virial route pressure,

```
#!/usr/bin/env python3
from oz import wizard as w
w.initialise()
w.arep[0,0] = A = 25.0
w.dpd_potential()
w.rho[0] = rho = 3.0
w.hnc_solve()
print('rho =', rho, ' A =', A)
print('pressure =', w.press)
print('energy density =', w.uex)
```

with the result

```
rho = 3.0  A = 25.0
pressure = 23.564147563638308
energy density = 13.761952448736555
```

The first line imports the `wizard` of `oz` and renames it `w`. The `oz` package and the `wizard` module are automatically generated by `f2py`. The actual results from accurate Monte-Carlo simulations are $p = 23.71 \pm 0.01$ and $e = 13.83 \pm 0.01$, so the HNC virial route result is in error by $\approx 0.5\%$.

A virtue of `python` is the large package library, for instance for graphical output. The following code uses the `matplotlib` package to plot the pair distribution function and structure factor (with a standard normalisation),

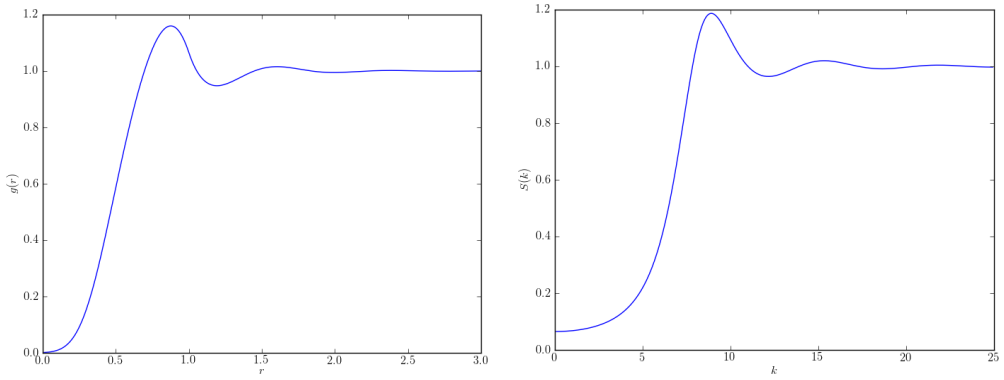

Figure 1: HNC pair distribution function and structure factor for standard DPD at $\rho = 3$ and $A = 25$.

```
#!/usr/bin/env python3
import matplotlib.pyplot as plt
from oz import wizard as w
w.initialise()
w.arep[0,0] = A = 25.0
w.dpd_potential()
w.rho[0] = rho = 3.0
w.hnc_solve()
plt.figure(1)  # This will be g(r)
imax = int(3.0 / w.deltar)
plt.plot(w.r[0:imax], 1.0 + w.hr[0:imax,0,0])
plt.xlabel('$r$')
plt.ylabel('$g(r)$')
plt.figure(2)  # This will be S(k)
jmax = int(25.0 / w.deltak)
plt.plot(w.k[0:jmax], w.sk[0:jmax,0,0]/rho)
plt.xlabel('$k$')
plt.ylabel('$S(k)$')
plt.show()
```

The result is shown in Fig. 1 (imagine trying to do this in pure FORTRAN!).

6.2 Compressibility route pressure

The following `python` routine integrates the compressibility along an isotherm to calculate the compressibility route pressure,

```
def cr_press(drho):
    p_xc = prev = 0.0
    n = int(w.rho[0]/drho + 0.5)
    w.cold_start = 1
    for i in range(n):
        w.rho[0] = drho * (i + 1.0)
        w.hnc_solve()
        p_xc = p_xc + 0.5 * drho * (prev + w.comp_xc)
        prev = w.comp_xc
    return w.rho[0] + p_xc
```

Points to note are that a trapezium rule is used for the numerical integration, and the routine actually integrates the excess compressibility since the ideal contribution to the pressure is trivially ρ.

When the `cr_press` routine finishes, the system is in the same state as it started, with the thermodynamics completely solved. So for example

```
w.initialise()
w.arep[0,0] = 25.0
w.dpd_potential()
w.rho[0] = 3.0
print('CR pressure =', cr_press(0.05))
print('VR pressure =', w.press)
```

results in

```
CR pressure = 22.6662332442
VR pressure = 23.5641475636383
```

Pressures calculated by the two routes differ by about 4% which appears to be typical for HNC for soft potentials. Also, further investigation reveals the dependence on $\delta\rho$ is linear and extrapolates to $p = 22.31$ at $\delta\rho \to 0$, for $\rho = 3$ and $A = 25$. So the error from discretising the isotherm is here $\lesssim 2\%$.

6.3 Free energy

One can calculate the pressure *via* the energy route, but this really involves the calculation of the free energy. One standard way this can be done is by coupling constant integration. The basic result can be derived by differentiating the fundamental expression $e^{-\beta A^{\rm ex}} = \int d\Omega\, e^{-\beta\phi}$ with respect to a

parameter in the potential ϕ. For instance for standard single-component DPD, $\partial a^{\rm ex}/\partial A = u^{\rm ex}/A$ where U is the internal energy. Thus

$$\beta a^{\rm ex} = \int_0^A \frac{dA'}{A'}\, \beta u^{\rm ex}_{A'} \tag{78}$$

can be evaluated along an isochore (constant density line). The following `python` routine uses this to compute the excess free energy

```
def energy_aex(dA):
    aex_xc = prev = 0.0
    n = int(w.arep[0,0]/dA + 0.5)
    w.cold_start = 1
    for i in range(n):
        w.arep[0,0] = dA * (i + 1.0)
        w.dpd_potential()
        w.hnc_solve()
        curr = w.uex_xc / w.arep[0,0]
        aex_xc = aex_xc + 0.5*dA*(prev + curr)
        prev = curr
    return w.uex_mf + aex_xc
```

Points to note are again that this implements a trapezium rule, and that the actual integration is carried out for the correlation contribution for which $u^{\rm ex}/A$ vanishes as $A \to 0$. The mean field contribution to the internal energy is strictly proportional to A, so it passes unscathed through Eq. (78) to contribute additively to the free energy.

For the HNC specifically, we could also calculate the free energy by integrating $\mu^{\rm ex}$ along an isotherm, giving a second routine

```
def mu_aex(drho):
    aex = prev = 0.0
    n = int(w.rho[0]/drho + 0.5)
    w.cold_start = 1
    for i in range(n):
        w.rho[0] = drho * (i + 1.0)
        w.hnc_solve()
        aex = aex + 0.5*drho*(prev + w.muex[0])
        prev = w.muex[0]
    return aex
```

Both routines leave the system in the same state as it started. As an example, to test these,

```
w.initialise()
w.rho[0] = 3.0
w.arep[0,0] = 25.0
w.dpd_potential()
print('energy aex =', energy_aex(0.1))
print('mu aex     =', mu_aex(0.05))
print('direct aex =', w.aex)
```

(the last of these uses the built-in free energy evaluation for HNC), giving

```
energy aex = 15.9476521676
mu aex     = 15.9481839257
direct aex = 15.947542189636641
```

We see that all three routes agree to the third decimal place, which is a good test of numerical discretisation error (the absolute HNC deficit is 2.6×10^{-4}). The above examples in sections 6.1–6.3 have been coded up as `examples.py`.

6.4 Further examples

See comments in files for more details.

- `gw_p_compare.py` – Plot $(p - \rho)/A\rho^2$ versus ρ, and compare with data from Fig. 4 of Ref. [12]. The agreement is very good.

- `wsg_fig5_compare.py` – Compute the free energy difference between a system of $N-1$ A particles and a B particle, and a pure system of N A particles, as a function of Δa, where $A_{00} = A_{11} = a$, $A_{01} = a + \Delta a$, and $a = 25$. This concept was introduced in Ref. [13] and is representative of an effective χ value. Here, the free energy change is calculated as $\mu_1^{\text{ex}} - \mu_0^{\text{ex}}$ in a two component problem with the mole fraction of the second species set to zero ($\rho_0 = 3$ and $\rho_1 = 0$). The results are compared with data from Fig. 5 of Ref. [13], and with independently done Monte-Carlo simulations using trial particle identity swaps.

- `vlm_fig2_compare.py` – Compute $\log_{10} \gamma_\infty = (\mu_1^{\text{ex}} - \mu_0^{\text{ex}})/\ln 10$ in the infinite dilution limit for a binary DPD fluid, and compare to the empirical fit $0.144 \times \Delta a$ as in Fig. 2 in Ref. [14]. The fit is derived by these authors from Monte-Carlo trial particle (Widom) insertions. This is essentially the same calculation as the preceding example except that $a = 106.5$. For such a large base repulsion amplitude, HNC doesn't converge from a cold start, and this example illustrates the trick of warming up the solver by ramping a up in stages.

- `x_dmu_compare.py` – In an unpublished extension to the above two examples $\mu_1^{\rm ex} - \mu_0^{\rm ex}$ is plotted as a function of the mole fraction x in a mixture where $\rho_0 = (1-x)\rho$ and $\rho_1 = x\rho$, for $a = 25$ and $\Delta a = 5$ (one cannot have Δa too large otherwise an underlying demixing transition causes HNC to fail). The results are compared to data from Monte-Carlo (MC) trial particle identity swaps. The agreement between HNC and MC is excellent, including the small deviation from linearity. The near-exact linear dependence can be represented by $\chi(1-2x)$ where χ is the Flory χ-parameter.

- `hm_fig4-2_compare.py` solves for the virial and compressibility route pressure for the MSA and HNC, for hard spheres, and compares with the accurate Carnahan-Starling equation of state, as in Fig 4.2 of Ref. [3].

- `hm_fig10-2_compare.py` solves the MSA and HNC for the RPM and compares with Monte-Carlo data, as in Fig 10.2 of Ref. [3].

- `attard_fig1_compare.py` solves the HNC for the RPM to generate plots of total correlation functions like that shown in Ref. [15].

- `urpm_oneoff.py`, `urpm_scan.py` solve the ultrasoft restricted primitive model (URPM) [1], including the possibility of a neutral solvent species. The first can be used for one-off calculations of structure and thermodynamics at a chosen state point. The second is intended to scan a range of densities to locate the Kirkwood line (transition between pure exponential and damped oscillatory behaviour in the tails of $h_{\mu\nu}$). Both routines are driven by command line options, via the `argparse` module. To see the available options use `--help`.

- `rpm_oneoff.py`, `hs_oneoff.py` are similar drivers for the RPM and hard sphere systems. Use `--show` in these to get a graphical output.

- `driver1-4.f90`, `driver3.py` are retained for regression testing.

- `fftw_test.f90` is test code for the fast Fourier transform library.

A Appendix

A.1 MSA charge-charge structure factor for RPM

The MSA solution for the RPM is known analytically [16]. Some results are summarised here, for benchmarking the code. We focus particularly on the

charge-charge structure factor as the closed form solution is fairly simple. As in section 3.3 we suppose the potential is given by

$$\beta v_{ij} = \begin{cases} \infty & r \leq R \\ z_i z_j l_{\rm B}/r & r > R \end{cases} \tag{79}$$

where $z_1 = 1$ and $z_2 = -1$. In this section we shall use R for the hard core diameter, to match with literature notation. We shall set $\rho_1 = \rho_2 = \rho/2$. Generically, because of symmetry ($h_{11} = h_{22}$ and $h_{12} = h_{21}$ *et c.*), the charge-charge structure factor is given by

$$S_{ZZ} = 1 + \tfrac{1}{2}\rho(\tilde{h}_{11} - \tilde{h}_{12})\,. \tag{80}$$

The symmetry also means that the OZ equation decouples,

$$\tilde{h}_{11} - \tilde{h}_{12} = \tilde{c}_{11} - \tilde{c}_{12} + \tfrac{1}{2}\rho(\tilde{h}_{11} - \tilde{h}_{12})(\tilde{c}_{11} - \tilde{c}_{12})\,. \tag{81}$$

This means that the charge-charge structure factor can be expressed simply in terms of the Fourier-transformed direct correlation functions,

$$S_{ZZ} = [1 - \tfrac{1}{2}\rho(\tilde{c}_{11} - \tilde{c}_{12})]^{-1}\,. \tag{82}$$

Now we turn to the MSA solution of the RPM [16]. There are two ways to calculate the charge-charge structure factor, both of which lead to the same result. The first is to Fourier transform the known solution to the real space direct correlation function. For this, we introduce the inverse Debye length $\kappa = (4\pi l_{\rm B}\rho)^{1/2}$, and dimensionless version of the same, $x = \kappa R$. In terms of the latter we define $B = [1+x-(1+2x)^{1/2}]/x$. Then the MSA solution for the direct correlation function difference is $c_{11} - c_{12} = -(4l_{\rm B}/R)(B - B^2 r/2R)$ for $r \leq R$, and $c_{11} - c_{12} = -2l_{\rm B}/r$ for $r > R$ (the latter reflects the MSA choice $c_{\mu\nu} = -\beta v_{\mu\nu}$ outside the hard core). The reason for focusing on the charge-charge structure factor is now apparent as this difference expression does not involved the Percus-Yevick solution for uncharged hard spheres, which would otherwise accompany the individual direct correlation functions.

To do the Fourier transform we write this as

$$\tfrac{1}{2}(c_{11} - c_{12}) = -\frac{l_{\rm B}}{r} + \Big[\frac{l_{\rm B}}{r} - \frac{2l_{\rm B}}{R}\Big(B - \frac{B^2 r}{2R}\Big)\Big]\,\Theta(R - r) \tag{83}$$

then, *c. f.* Eq. (2),

$$\tfrac{1}{2}(\tilde{c}_{11} - \tilde{c}_{12}) = -\frac{4\pi l_{\rm B}}{k^2} + \frac{4\pi}{k}\int_0^R dr\, r\, \sin kr\, \Big[\frac{l_{\rm B}}{r} - \frac{2l_{\rm B}}{R}\Big(B - \frac{B^2 r}{2R}\Big)\Big]\,. \tag{84}$$

The integral is easily evaluated and inserted in the expression for the charge-charge structure factor to obtain, after some simplification,

$$S_{ZZ}(k) = \frac{k^4}{k^4 + 8q^4 + 4q^2(k^2 - 2q^2)\cos kR + 8kq^3 \sin kR} \,. \tag{85}$$

In this, $q = [(1+2x)^{1/2} - 1]/(2R)$.[2] Eq. (85) can be compared directly with the charge-charge structure factor reported by `rpm_oneoff.py`. Complete agreement to within numerical accuracy should be observed. Note that correctly $S_{ZZ}(k) = k^2/\kappa^2 + O(k^4)$ as $k \to 0$ (thus the MSA satisfies the Stillinger-Lovett moment conditions), and $S_{ZZ}(k) \to 1$ as $k \to \infty$.

For completeness, the partner to Eq. (83) is the Percus-Yevick hard sphere solution [3, 17]

$$\tfrac{1}{2}(c_{11} + c_{12}) = -[\lambda_1 - 6\eta\lambda_2 r/R + \tfrac{1}{2}\eta\lambda_1(r/R)^3]\,\Theta(1-x) \tag{86}$$

where $\lambda_1 = (1+2\eta)^2/(1-\eta)^4$, $\lambda_2 = (1+\frac{1}{2}\eta)^2/(1-\eta)^4$, and $\eta = \pi\rho R^3/6$ is the packing fraction.

An alternative route to the charge-charge structure factor uses analytic continuation on the Laplace-transformed direct correlation function. Specifically, Waisman and Lebowitz [16] define $\bar{G}(s) = \int_0^\infty dr\, e^{-sr} r\,[g_{11}(r) - g_{12}(r)]$ and give the result (see also Stell and Sun [18])

$$\bar{G}(s) = -s(2q^2/\pi\rho)[(s^2 + 2qs + 2q^2)e^{sR} - 2q^2]^{-1} \,. \tag{87}$$

From the definitions it is easy to infer

$$S_{ZZ}(k) = 1 + \frac{2\pi\rho}{k}\,\mathrm{Im}\,\bar{G}(-ik) \,. \tag{88}$$

If the indicated operations are carried out, one indeed recovers Eq. (85).

References

[1] P. B. Warren, A. Vlasov, L. Anton and A. J. Masters "Screening properties of Gaussian electrolyte models, with application to dissipative particle dynamics", J. Chem. Phys. **138**, 204907 (2013).

[2] K.-C. Ng, "Hypernetted chain solutions for the classical one-component plasma up to $\Gamma = 7000$", J. Chem. Phys. **61**, 2680–2689 (1974).

[2] Thus $x = 2qR(1+qR)$, and $B = qR/(1+qR)$.

[3] J.-P. Hansen and I. R. McDonald, "Theory of simple liquids 3rd edition" (Academic Press, Amsterdam, 2006).

[4] C. T. Kelley and B. Montgomery Pettitt, "A fast solver for the Ornstein-Zernike equations", J. Comp. Phys. **197**, 491–501 (2004). Beware typos in this paper! Note, i and j in the main text are $i-1$ and $j-1$ in this paper, and $n = N-1$. This paper also suggests that c and e at $r = 0$ are evaluated by simple linear extrapolation, $c_0 = 2c_1 - c_2$ and $e_0 = 2e_1 - e_2$; and c and e at $r = n\delta_r$ are zero, $c_n = e_n = 0$.

[5] L. Vrbka, M. Lund, I. Kalcher, J. Dzubiella, R. R. Netz, and W. Kunz, "Ion-specific thermodynamics of multicomponent electrolytes: A hybrid HNC/MD approach", J. Chem. Phys. **131**, 154109 (2009).

[6] K. Hiroike, "A new approach to the theory of classical fluids, II", Prog. Theor. Phys. **24**, 317 (1960).

[7] L. Lue and D. Blankschtein, "Proper integral equations for interaction-site fluids: exact free-energy expressions", J. Chem. Phys. **100**, 3002 (1994).

[8] P. B. Warren and A. Vlasov "Screening properties of four mesoscale smoothed charge models, with application to dissipative particle dynamics", J. Chem. Phys. **140**, 084904 (2014).

[9] R. D. Groot, "Electrostatic interactions in dissipative particle dynamics—simulation of polyelectrolytes and anionic surfactants", J. Chem. Phys. **118**, 11265 (2003).

[10] M. González-Melchor, E. Mayoral, M. E. Velázquez and J. Alejandre, "Electrostatic interactions in dissipative particle dynamics using the Ewald sums", J. Chem. Phys. **125**, 224107 (2006).

[11] W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling, "Numerical recipes in FORTRAN 77: 2nd edition" (CUP, Cambridge, 1992).

[12] R. D. Groot and P. B. Warren, "Dissipative particle dynamics: bridging the gap between atomistic and mesoscopic simulation", J. Chem. Phys. **107**, 4423 (1997).

[13] C. M. Wijmans, B. Smit and R. D. Groot, "Phase behavior of monomeric mixtures and polymer solutions with soft interaction potentials", J. Chem. Phys. **114**, 7644 (2001).

[14] A. Vishnyakov, M. T. Lee and A. V. Neimark, "Prediction of the critical micelle concentration of nonionic surfactants by dissipative particle dynamics simulations", J. Phys. Chem. Lett. **4**, 797 (2013).

[15] P. Attard, "Asymptotic analysis of primitive model electrolytes and the electrical double layer", Phys. Rev. E **48**, 3604 (1993).

[16] E. Waisman and J. L. Lebowitz, "Mean spherical model integral equation for charged hard spheres. II Results", J. Chem. Phys. **56**, 3093 (1972).

[17] M. S. Wertheim, "Exact solution of the Percus-Yevick integral equation for hard spheres", Phys. Rev. Lett. **10**, 321 (1963).

[18] G. Stell and S. F. Sun, "Generalised mean spherical approximation for charged hard spheres: The electrolyte regime", J. Chem. Phys. **63**, 5333 (1975).