Webnucleo Technical Report: Screening and Reverse Rate Correction Factors in libnucnet

Bradley S. Meyer

June 5, 2016

This technical report describes how to provide routines to include screening and reverse ratio correction factors for rate calculations in libnucnet.

1 Screening

The dense electron gas present in plasmas can enhance the rate for a thermonuclear reaction because the negative charge of the electrons screens the positive charges of the interacting nuclei. This makes the penetration of the Coulomb barrier between the two positively charged reactants easier and thus increases their interaction rate. Expressions for screening exist in the literature and have been widely employed in nucleosynthesis calculations (e.g., [1]). The question addressed here is how to implement screening in libnucnet.

One way to do this is to compute the forward and reverse nuclear reaction rates for a zone using the libnucnet API routine Libnucnet__Zone__computeRates(). Once the rates are computed, one can then iterate over the reactions and apply the screening function to each reaction. Once the screening factor is computed, it is then applied to the rates for the reaction by using the API routine Libnucnet__Zone__updateRatesForReaction().

The libnucnet API allows for a more systematic treatment. From versions 0.2 to 0.25 of libnucnet, the user supplied a routine, called a Libnucnet__Net__screening_function in libnucnet. This then was called from an API routine Libnucnet__Net__computeScreeningFactorForReaction. This then required a separate application function. As of version 0.26, this has all been simplified. To apply screening to rates in a particular zone, the user simply writes a Libnucnet__Zone__screeningFunction with prototype

```
void
Libnucnet__Zone__screeningFunction(
  Libnucnet__Zone * self,
  Libnucnet__Reaction * p_reaction,
  double d_t9,
  double d_rho,
  double d_ye,
```

```
  double * p_forward_rate,
  double * p_reverse_rate
);
```

The routine may have any appropriate name. The necessary inputs are:

- **self:** A pointer to the Libnucnet__Zone.
- **p_reaction:** A pointer to a Libnucnet__Reaction to which the screening is to be applied.
- **d_t9:** The temperature T_9, that is, the temperature in 10^9K at which to compute the screening factor.
- **d_rho:** The density ρ in g/cc at which to compute the screening factor.
- **d_ye:** The electron-to-baryon ratio Y_e at which to compute the screening factor.
- **p_forward_rate:** A pointer to the current value of the forward rate for the reaction.
- **p_reverse_rate:** A pointer to the current value of the reverse rate for the reaction.

Given these inputs, the user's routine should compute the screening factor for the reaction and apply it as appropriate to the forward and reverse rates.

With a properly defined zone screening function, the user then sets it and any applicable extra data with the API routine Libnucnet__Zone__setScreeningFunction(), which has the prototype

```
void
Libnucnet__Zone__setScreeningFunction(
  Libnucnet__Zone *self,
  Libnucnet__Zone__screeningFunction my_zone_screening_function,
  void *p_user_data
);
```

The inputs to this routine are

- **self:** A pointer to the Libnucnet__Zone under consideration.
- **my_zone_screening_function:** The name of the user's screening function. The user should cast this as a Libnucnet__Zone__screeningFunction.
- **p_my_data:** The pointer to the user's data structure. If there are no extra data to pass to the user's function, this should be NULL.

Once the screening function and data have been set for a zone, that function will be called for all reactions when Libnucnet__Zone__computeRates() is called. To retrieve the data for the screening function set for the zone, one calls the API routine Libnucnet__Zone__getScreeningData. To clear the screening function, the user simply calls the API routine Libnucnet__Zone__clearScreeningFunction().

The example screening function in the libnucnet distribution computes the screening factor for reactions with more than two reactants in the traditional manner of viewing the full reaction as a sequence of intermediate reactions. For example, for the three-body reaction $a + b + c$, libnucnet uses the user's screening routine to compute $F_{screen}(a, b)$, the screening factor for the reaction $a + b$. It then uses the user's screening routine to compute $F_{screen}(a + b, c)$, the screening factor for the reaction $(a + b) + c$. The total screening factor applied to the reaction $a + b + c$ is then $F_{screen}(a, b) \times F_{screen}(a + b, c)$. For four reactants, that is, the reaction $a + b + c + d$, the total screening factor would be $F_{screen}(a, b) \times F_{screen}(a + b, c) \times F_{screen}(a + b + c, d)$. Libnucnet loops over reactants, so this generalizes to any number of reactants. As of version 0.26, the order in which the screening for the reactants is applied is simply the order in which the reactants are stored for the reaction. The user can certainly change this in his or her screening function.

In versions 0.24 and 0.25, libnucnet applied either the forward or reverse screening factor to both the forward and reverse rates, depending on a restricted set of rules. As of version 0.26, since the user has full freedom in defining how the screening factors are applied, a number of routines related to the version 0.24 and 0.25 treatments have been removed from the API. Nevertheless, the example screening code retains some of that treatmenet. In particular, it applies the greater of the forward or reverse screening factors to both the forward and reverse rates. Again, the user may change this in his or her screening function.

By multiplying both forward and reverse rates for a reaction by the computed screening factor, the default network will tend to evolve at constant temperature and density and in the absence of weak interaction rates to the same nuclear statistical equilibrium (NSE) that it would have without screening applied. This may not be consistent with the physical nature of the screening. In particular, the presence of the electrons may alter the binding energy of the nuclei and hence the resulting NSE. If the user wishes to correct the reverse reaction rate so that the NSE is consistent with the screening due to the electrons, he or she must apply the reverse ratio correction factor function as described in the next section.

2 Reverse Ratio Correction Factor

As discussed above, libnucnet example screening applies the result of the user's screening function to both the forward and reverse rates for a reaction. This means that the network abundances will tend to evolve to the same NSE that they would without application of the screening (albeit at a different overall rate). If this is not consistent with the user's screening model, he or she must

apply a correction factor to the reverse ratio. This correction factor is in fact derived from the factor by which one would multiply the NSE abundance for a species to account for the effect of the electrons (or another effect).

We begin by explaining what is meant by a libnucnet NSE correction factor. The condition for NSE is

$$\mu_i = Z_i \mu_p + N_i \mu_n, \tag{1}$$

where μ_i is the chemical potential for species i and μ_n, μ_p are the chemical potentials for neutrons and protons, respectively, and Z_i and N_i are the atomic number and neutron number for species i, respectively. The libnucnet default is to consider all nucleons and nuclei as ideal, classical Maxwell-Boltzmann particles; thus,

$$\mu_i = m_i c^2 + kT \ln \left(\frac{Y_i}{Y_{Qi}} \right) \equiv m_i c^2 + \mu_i', \tag{2}$$

where the quantum abundance Y_{Qi} is defined as

$$Y_{Qi} \equiv \frac{G_i}{\rho N_A} \left(\frac{m_i kT}{2\pi\hbar^2} \right)^{3/2}. \tag{3}$$

In Eq. (2) Y_i is the abundance of i per nucleon and $m_i c^2$ is the rest mass energy of species i. In Eq. (3) N_A is Avogadro's number, G_i is i's nuclear partition function, and k is Boltzmann's constant. It is useful to consider that Y_{Qi} is the abundance per nucleon of species i if there were one such particle in a cube with side one thermal de Broglie wavelength in length. The NSE abundance of species i is then

$$Y_{i,NSE} = Y_{Qi} \exp \left\{ Z_i \frac{\mu_p'}{kT} + N_i \frac{\mu_n'}{kT} + \frac{B_i}{kT} \right\}, \tag{4}$$

where B_i is the nuclear binding energy of species i:

$$B_i = Z_i m_p c^2 + N_i m_n c^2 - m_i c^2. \tag{5}$$

As of version 0.7 of libnucnet, the quantities Y_{Qi} and B_i can be computed from the API routines *Libnucnet__Species__computeQuantumAbundance()* and *Libnucnet__Nuc__computeSpeciesBindingEnergy()*, respectively.

The libnucnet NSE correction factor is a factor $f_{i,corr}$ that gets added to the exponent in Eq. (4) to correct for deviations away from the above treatment. In particular, suppose in the correct treatment, $\mu_i' \neq kT \ln (Y_i / Y_{Qi})$. We may then write

$$Y_i = Y_{Qi} \exp \left\{ \frac{\mu_i'}{kT} + \left[\ln \left(\frac{Y_i}{Y_{Qi}} \right) - \frac{\mu_i'}{kT} \right] \right\}. \tag{6}$$

If we now apply the NSE condition in Eq. (1), we find

$$Y_{i,NSE}' = Y_{Qi} \exp \left\{ Z_i \frac{\mu_p'}{kT} + N_i \frac{\mu_n'}{kT} + \frac{B_i}{kT} + \left[\ln \left(\frac{Y_i}{Y_{Qi}} \right) - \frac{\mu_i'}{kT} \right] \right\}, \tag{7}$$

where $Y'_{i,NSE}$ indicates the nuclear statistical equilibrium abundance in the now correct treatment. We thus see that

$$Y'_{i,NSE} = Y_{i,NSE} \exp\left\{\ln\left(\frac{Y_i}{Y_{Qi}}\right) - \frac{\mu'_i}{kT}\right\} \equiv Y_{i,NSE} \times e^{f_{i,corr}},, \tag{8}$$

where $Y_{i,NSE}$ refers to the equilibrium abundance computed from the neutron and proton chemical potentials and the species binding energy.

An example will illustrate how this works. Suppose each nuclear species has a constant potential energy U_i, perhaps due to a uniform background of electrons. The abundance of the species is then given by

$$Y_i = Y_{Qi} \exp\left\{\frac{\mu'_i}{kT} - \frac{U_i}{kT}\right\}. \tag{9}$$

From this, it is clear that

$$f_{i,corr} = -\frac{U_i}{kT}. \tag{10}$$

To accommodate an NSE correction factor, the user first writes a Libnucnet__Species__nseCorrectionFactorFunction() with any appropriate name (in the case below my_correction_function) which has the prototype

```
double
my_correction_function(
  Libnucnet__Species *self,
  double d_t9,
  double d_rho,
  double d_ye,
  void *p_my_data
);
```

The inputs are

- **self:** A pointer to a Libnucnet__Species.
- **d_t9:** The temperature T_9, that is, the temperature in 10^9 K at which to compute the correction factor.
- **d_rho:** The density ρ in g/cc at which to compute the correction factor.
- **d_ye:** The electron-to-baryon ratio Y_e at which to compute the correction factor.
- **p_my_data:** A pointer to a user-defined data structure containing any extra data to be passed into the user's routine. This should be NULL if no extra data are required.

This function has the Species namespace because it generally requires only nuclear data for a particular species, along with the temperature, density, and Y_e. The output from this routine is a double that contains the natural logarithm of the factor by which one multiplies the NSE abundance of a species computed from the proton and neutron chemical potentials and the species binding energy to account for the correction. Thus, in the case of electron screening, if the abundance of species i in NSE in the absence of the electron effects is $Y_{i,NSE}$, the user's Libnucnet__Species__nseCorrectionFactorFunction should return the quantity $f_{i,corr}$ such that the NSE abundance with the effect of the electrons included is $\exp(f_{i,corr}) \times Y_{i,NSE}$.

Once the species NSE correction factor function is defined, the user may compute the reverse ratio correction factor for a reaction by calling Libnucnet__Net__computeReverseRatioCorrectionFactorForReaction() directly. The prototype for this reaction is:

```
double
Libnucnet__Net__computeReverseRatioCorrectionFactorForReaction(
  Libnucnet__Net *self,
  Libnucnet__Reaction *p_reaction,
  double d_t9,
  double d_rho,
  double d_ye,
  Libnucnet__Species__nseCorrectionFactorFunction
    my_correction_function,
  void *p_my_data
);
```

The inputs are

- **self:** A pointer to a Libnucnet__Net structure, which contains the nuclear network (nuclear + reaction) data.
- **p_reaction:** A pointer to the Libnucnet__Reaction structure for the reaction whose reverse ratio correction factor is desired.
- **d_t9:** The temperature T_9, that is, the temperature in 10^9 K at which to compute the correction factor.
- **d_rho:** The density ρ in g/cc at which to compute the correction factor.
- **d_ye:** The electron-to-baryon ratio Y_e at which to compute the correction factor.
- **my_correction_function:** The name of the user's correction factor function. The user should cast this as a Libnucnet__Species__nseCorrectionFactorFunction.
- **p_my_data:** A pointer to a user-defined data structure containing any extra data to be passed into the user's routine. This should be NULL if no extra data are required.

The routine loops over all reactants and products in the reaction and computes their NSE correction factors. From these, the routine then returns the double by which the reverse reaction rate should be multiplied to account for the electron screening effects.

The user may set a species correction factor function for a zone by calling Libnucnet__Zone__setNseCorrectionFactorFunction(), which has the prototype

```
void
Libnucnet__Zone__setNseCorrectionFactorFunction(
  Libnucnet__Zone *self,
  Libnucnet__Species__nseCorrectionFactorFunction
    my_correction_function,
  void *p_my_data
);
```

The inputs to this routine are

- **self:** A pointer to the Libnucnet__Zone under consideration.
- **my_correction_function:** The name of the user's correction factor function. The user should cast this as a Libnucnet__Species__nseCorrectionFactorFunction.
- **p_my_data:** The pointer to the user's data structure. If there are no extra data to pass to the user's function, this should be NULL.

As of version 0.26, once the species NSE correction factor function is set for a zone, a user can retrieve it with the API routine Libnucnet__Zone__getNseCorrectionFactorFunction(). One can also retrieve the associated data with Libnucnet__Zone__getNseCorrectionFactorData().

In versions 0.24 and 0.25, libnucnet applied the reverse ratio correction factor to reactions separately from the screening. As of version 0.26, application of the reverse ratio correction factor to reactions is entirely up to the user. In the example screening with libnucnet 0.26 and later, the reverse ratio correction factor is applied within the screening function. Users should consult the example codes to see how that works.

3 Corrections and NSE

In order to be clear about the procedure for applying the screening and reverse ratio correction factors, we provide an example. Consider the reaction

$$^{12}\mathrm{C} + \alpha \rightarrow ^{16}\mathrm{O} + \gamma. \tag{11}$$

The contribution of this reaction to $dY(^{12}\mathrm{C})/dt$, the time rate of change of the abundance of $^{12}\mathrm{C}$, is

$$-N_A\langle\sigma v\rangle\rho Y(^{12}\mathrm{C})Y(\alpha) + \lambda_\gamma Y(^{16}\mathrm{O}), \tag{12}$$

where N_A is Avogadro's number, $\langle\sigma v\rangle$ is the thermally-averaged interaction cross section for a reaction between a $^{12}\mathrm{C}$ nucleus and an α particle, ρ is the

mass density, and λ_γ is the rate per second for an ^{16}O nucleus to disintegrate back into a ^{12}C and an α.

If the system attains NSE, then detailed balance ensures that forward and reverse reaction flows come into balance. Thus,

$$N_A\langle\sigma v\rangle\rho Y_{NSE}(^{12}\mathrm{C})Y_{NSE}(\alpha) = \lambda_\gamma Y_{NSE}(^{16}\mathrm{O}), \tag{13}$$

This means that the "reverse ratio" R relating the forward and reverse reaction rates is

$$R = \frac{\lambda_\gamma}{N_A\langle\sigma v\rangle} = \frac{\rho Y_{NSE}(^{12}\mathrm{C})Y_{NSE}(\alpha)}{Y_{NSE}(^{16}\mathrm{O})}. \tag{14}$$

In this particular case, we can apply Eq. (4) to find

$$R = \rho\frac{Y_{Q,^{12}\mathrm{C}}Y_{Q,\alpha}}{Y_{Q,^{16}\mathrm{O}}}\exp\left\{\frac{B_{^{12}\mathrm{C}}}{kT} + \frac{B_\alpha}{kT} - \frac{B_{^{16}\mathrm{O}}}{kT}\right\}. \tag{15}$$

If we compute reverse rates from forward rates using reverse ratios computed from detailed balance in this way, our network will tend to evolve at constant temperature and density and in the absence of weak interaction rates to the expected NSE.

We now compute screening. Suppose the screening factor for the forward reaction in Eq. (11) is $F_{screen}(\alpha,\gamma)$ and for the reverse reaction is $F_{screen}(\gamma,\alpha)$. If we apply these screening factors, we find

$$R' = \frac{F_{screen}(\gamma,\alpha)\lambda_\gamma}{F_{screen}(\alpha,\gamma)N_A\langle\sigma v\rangle} = \frac{F_{screen}(\gamma,\alpha)}{F_{screen}(\alpha,\gamma)} \times R. \tag{16}$$

Since libnucnet computes a reverse rate from a forward rate using detailed balance, the effect of screening on the reverse rate is obtained from consideration of the NSE achieved. As mentioned above, the user-supplied NSE correction factor function computes the factor by which $Y_{i,NSE}$, the NSE abundance of species i, changes due to the screening electrons; thus, if the user's routine returns the correction factor $f_{i,corr}$ for species i, then

$$Y'_{i,NSE} = e^{f_{i,corr}} \times Y_{i,NSE}, \tag{17}$$

where the prime indicates correction for the screening electrons. Because of detailed balance, we thus know

$$R' = \frac{\rho Y'_{NSE}(^{12}\mathrm{C})Y'_{NSE}(\alpha)}{Y'_{NSE}(^{16}\mathrm{O})} = e^{f_{^{12}\mathrm{C},corr}+f_{\alpha,corr}-f_{^{16}\mathrm{O},corr}} \times \frac{\rho Y_{NSE}(^{12}\mathrm{C})Y_{NSE}(\alpha)}{Y_{NSE}(^{16}\mathrm{O})} \tag{18}$$

Comparison of Eqs. (14), (16), and (18) shows that

$$F_{screen}(\gamma,\alpha) = e^{f_{^{12}\mathrm{C},corr}+f_{\alpha,corr}-f_{^{16}\mathrm{O},corr}} \times F_{screen}(\alpha,\gamma). \tag{19}$$

We may write

$$R' = F_{corr} \times R. \tag{20}$$

F_{corr} is the reverse ratio correction factor, the quantity returned by

Libnucnet__Net__computeReverseRatioCorrectionFactorForReaction()

For the reaction in Eq. (11), it is clear that

$$F_{corr} = e^{f_{^{12}\mathrm{C},corr} + f_{\alpha,corr} - f_{^{16}\mathrm{O},corr}}. \tag{21}$$

In general,

$$F_{screen}(\text{reverse}) = F_{corr} \times F_{screen}(\text{forward}). \tag{22}$$

Libnucnet applies the user's screening function, that is, his or her

Libnucnet__screening_function()

to compute the screening factor for the forward rate and then the user's

Libnucnet__Species__nseCorrectionFactorFunction()

on the various reactants to compute the reverse ratio correction factor and the screening factor for the reverse reaction. With this treatment, the system will evolve in the absence of weak reactions and at constant temperature and density to the NSE appropriate to the user's NSE correction factor function [that is, once the abundances reach NSE, they will be given by Eq. (17)]. Note that if the user does not supply an NSE correction factor function, the forward and reverse ratios have the same screening factor. This means the abundances will achieve the same NSE they would have in the absence of screening although the time to reach that equilibrium would typically be shorter.

4 Turning off Reverse Rates

Some users may find it desirable to turn off computation of reverse rates by detailed balance. For example, a user may wish instead to supply his or her own forward and reverse rates. While this can be done by updating the reverse rate to zero after computation of all the rates for a zone, this is an inefficient procedure. As of version 0.18, it is possible to turn off automatic computation of reverse rates by detailed balance for a zone with the API routine Libnucnet__Zone__toggleReverseRateDetailedBalance(). The user supplies a pointer to a libnucnet zone and sets the switch to either the string "on" or "off". If the user sets the switch to "off", the libnucnet API routine Libnucnet__Zone__computeRates() will compute only the forward rate for every valid reaction, and the reverse rate will be zero. As of version 0.26, if the user does not compute reverse reactions from detailed balance, any zone screening function should apply the screening factor only to the forward reaction.

A user who toggles off computation of reverse rates by detailed balance must be careful to supply both the forward and reverse rates for a reaction separately.

For example, if the computation of reverse rates by detailed balance is on, it is sufficient to supply the rate data for the reaction $^{12}C +^{4} He \rightarrow^{16} O + \gamma$. libnucnet will automatically compute the rates for the forward reaction $^{12}C +^{4} He \rightarrow^{16} O + \gamma$ and the reverse reaction $^{16}O + \gamma \rightarrow^{12} C +^{4} He$ for a zone when Libnucnet__Zone__computeRates() is called. If, however, the computation of reverse rates by detailed balance is off, it is necessary to supply rate data for both the reaction $^{12}C +^{4} He \rightarrow^{16} O + \gamma$ and the reaction $^{16}O + \gamma \rightarrow^{12} C +^{4} He$ separately. The routine Libnucnet__Zone__computeRates() will compute the rates for both of these reactions as forward rates.

References

[1] R. K. Wallace, S. E. Woosley, and T. A. Weaver, *The thermonuclear model for x-ray transients*, Astrophys. J, 258 (1982), pp. 696–715.