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

Bradley S. Meyer

July 30, 2008

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__replaceRatesForReaction().

As of version 0.2 of libnucnet, there is a more efficient way to include screening in libnucnet calculations. The user may supply a routine, called a Libnucnet__Net__screeningFunction in libnucnet. Libnucnet will use this routine to compute screening.

To supply a screening function, the user first writes a routine with the prototype:

```
double
my_screening_function(
  double d_t9,
  double d_rho,
  double d_ye,
  int i_Z1,
  int i_A1,
  int i_Z2,
```

```
  int i_A2
  void *p_my_data
);
```

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

- **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.
- **i_Z1:** The atomic number of the first reactant.
- **i_A1:** The mass number of the first reactant.
- **i_Z2:** The atomic number of the second reactant.
- **i_A2:** The mass number of the second reactant.
- **p_my_data:** A pointer to a user-defined data structure containing extra data to be passed to the user's screening function. If there are no extra data, this should be set to NULL.

Given these inputs, the user's routine should return the factor by which to change the rate for the reaction $(i\_Z1, i\_A1) + (i\_Z2, i\_A2)$ to account for screening. Thus, if the rate for the reaction is [12] in the absence of screening, the rate for the reaction with screening should be $f \times$ [12], where f is the screening factor returned by the user's routine.

There are two ways to call the screening routine. The first way is to call the routine through the libnucnet API routine Libnucnet__Net__computeScreeningFactorForReaction(). The prototype for this routine is

```
double
Libnucnet__Net__computeScreeningFactorForReaction(
  Libnucnet__Net *self,
  Libnucnet__Reaction *p_reaction,
  double d_t9,
  double d_rho,
  double d_ye,
  Libnucnet__Net__screeningFunction my_screening_function,
  void *p_user_data
);
```

The necessary 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 screening factor is desired.
- **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.
- **my_screening_function:** The name of the user's screening function. Typically, this needs to be cast as a Libnucnet__Net__screeningFunction.
- **p_my_data:** A pointer to a user-define data structure containing extra data to be passed to the user's screening function. If there are no extra data, this should be NULL.

The output from this routine is a double containing the screening factor for the reaction pointed to by p_reaction, as computed by the user's routine. To apply this screening factor, the user then multiplies the forward and reverse rates for the reaction by the value returned by Libnucnet__Net__computeScreeningFactorForReaction().

A more useful alternative for network calculations is to set the screening function for a particular zone and then let Libnucnet__Zone__computeRates() call the user's routine directly. To do this, the user calls the API routine Libnucnet__Zone__setScreeningFunction(), which has the prototype

```
void
Libnucnet__Zone__setScreeningFunction(
  Libnucnet__Zone *self,
  Libnucnet__screeningFunction my_screening_function,
  void *p_user_data
);
```

The inputs to this routine are

- **self:** A pointer to the Libnucnet__Zone under consideration.
- **my_screening_function:** The name of the user's screening function. The user should cast this as a Libnucnet__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.

The user must set the screening function prior to calling Libnucnet__Zone__computeRates(), otherwise the screening function will not be called. To clear the screening function, the user simply calls the API routine Libnucnet__Zone__clearScreeningFunction().

Libnucnet 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.

After the screening factor f_{screen} is computed for the forward reaction, it is applied to both the forward and reverse reaction rates; thus, both the forward and reverse rates are multiplied by f_{screen}. It is possible to retrieve the screening factor applied to a reaction in a particular zone by calling the API routine Libnucnet__Zone__getScreeningFactorForReaction().

By multiplying both forward and reverse rates for a reaction by the computed screening factor, the 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 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 the factor by which one would multiply the NSE abundance for a species to account for the effect of the electrons.

To do this, 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 factor by which one multiplies the NSE abundance of a species to account for the electrons. Thus, 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_{corr} such that the NSE abundance with the effect of the electrons included is $f_{corr} \times Y_{i,NSE}$.

As with the screening routine, the user may call his or her routine in two ways. First, the user may call the libnucnet API routine 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 second way to allow libnucnet to call the user-supplied correction factor routine is to set the correction factor for a given zone and let the Libnucnet__Zone__computeRates() routine call it. To do this, the user calls the routine 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.

The routine uses the user's routine to compute the reverse ratio correction factor R and applies it to the reverse rate for each reaction.

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

$$^{12}\mathrm{C} + \alpha \rightarrow^{16} \mathrm{O} + \gamma. \tag{1}$$

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

$$-N_A\langle\sigma v\rangle\rho Y(^{12}\mathrm{C})Y(\alpha) + \lambda_\gamma Y(^{16}\mathrm{O}), \tag{2}$$

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 $^{16}\mathrm{O}$ nucleus to disintegrate back into a $^{12}\mathrm{C}$ and an α.

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

$$N_A\langle\sigma v\rangle\rho Y_{NSE}(^{12}\mathrm{C})Y_{NSE}(\alpha) = \lambda_\gamma Y_{NSE}(^{16}\mathrm{O}), \tag{3}$$

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

$$R = \frac{\lambda_\gamma}{N_A\langle\sigma v\rangle} = \frac{\rho Y_{NSE}(^{12}\mathrm{C})Y_{NSE}(\alpha)}{Y_{NSE}(^{16}\mathrm{O})}. \tag{4}$$

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. (1) is $f_{screen}(\alpha,\gamma)$ and for the reverse reaction is $f_{screen}(\gamma,\alpha)$. If we apply these screening factors, we find

$$R' = \frac{f_{screen}(\gamma,\alpha)\lambda_\gamma}{f_{screen}(\alpha,\gamma)N_A\langle\sigma v\rangle} = \frac{f_{screen}(\gamma,\alpha)}{f_{screen}(\alpha,\gamma)} \times R. \tag{5}$$

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 r_i for species i, then

$$Y'_{i,NSE} = r_i \times Y_{i,NSE}, \tag{6}$$

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

$$R' = \frac{\rho Y'_{NSE}(^{12}\mathrm{C})Y'_{NSE}(\alpha)}{Y'_{NSE}(^{16}\mathrm{O})} = \frac{r(^{12}\mathrm{C})r(\alpha)}{r(^{16}\mathrm{O})} \times \frac{\rho Y_{NSE}(^{12}\mathrm{C})Y_{NSE}(\alpha)}{Y_{NSE}(^{16}\mathrm{O})} \tag{7}$$

Comparison of Eqs. (4), (5), and (7) shows that

$$f_{screen}(\gamma,\alpha) = \frac{r(^{12}\mathrm{C})r(\alpha)}{r(^{16}\mathrm{O})} \times f_{screen}(\alpha,\gamma). \tag{8}$$

We may write

$$R' = f_{corr} \times R. \tag{9}$$

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

Libnucnet__Net__computeReverseRatioCorrectionFactorForReaction()

For the reaction in Eq. (1), it is clear that

$$f_{corr} = \frac{r(^{12}\mathrm{C})r(\alpha)}{r(^{16}\mathrm{O})} \tag{10}$$

In general,

$$f_{screen}(\text{reverse}) = f_{corr} \times f_{screen}(\text{forward}). \tag{11}$$

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

Libnucnet__screeningFunction()

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. (6)]. Note that if the user does not supply an NSE correction factor function, the forward and reverse ratios have the same screening factor (all r's will be unity). 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.

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.