Iver Warburton
ASTR 600 - Cosmology HW4
October 2023

1. Recombination

Saha equation for ionisation fraction X_e: $\left(\frac{1-X_e}{X_e^2}\right) = \frac{2\zeta(3)}{\pi^2}\eta\left(\frac{2\pi T}{m_e}\right)^{3/2} e^{E_I/T}$

a.) Derive this.

$$X_e = \frac{n_e}{n_p + n_H} \leftarrow \text{fraction of electrons left behind} \qquad n_H = n_p - n_n$$

$$1 - X_e = 1 - \frac{n_e}{n_p + n_H}$$

$$\frac{1-X_e}{X_e^2} = \frac{1 - \frac{n_e}{n_p+n_H}}{\frac{n_e^2}{(n_p+n_H)^2}}$$

$$= \frac{n_p + n_H - n_e}{n_p + n_H}\cdot\frac{(n_p+n_H)^2}{n_e^2}$$

$$= \frac{\cancel{n_p} + n_H - \cancel{n_e}}{n_e^2}(n_p + n_H), \text{ where Universe is uncharged so } n_p = n_e$$

$$\frac{1-X_e}{X_e^2} = \frac{n_H}{n_e^2}\overbrace{(n_p + n_H)}^{n_B}, \text{ where } \eta = \frac{n_B}{n_\gamma}$$

$$= \frac{n_H}{n_e^2}\eta n_\gamma, \text{ where } n_b = \eta n_\gamma = \eta \times \frac{2\zeta(3)}{\pi^2}T^3$$

$$= \eta\frac{2\zeta(3)}{\pi^2}T^3\frac{n_H}{n_e^2}$$

$$-E_I = m_H - m_p - m_e$$

$$\left(\frac{n_H}{n_e n_p}\right)_{eq} = \frac{g_H}{g_e g_p}\left(\frac{m_H}{m_e m_p}\frac{2\pi}{T}\right)^{3/2} e^{(m_p + m_e - m_H)/T}$$

$\hookrightarrow n_e = n_p$ (no charge); $\hookrightarrow m_H \simeq m_p$; $\hookrightarrow g_e = g_p = 2,\ g_H = 4 \rightarrow \frac{4}{2\cdot 2} = 1$; $(m_p + m_e - m_H) = E_I$

$$\left(\frac{n_H}{n_e^2}\right)_{eq} = \left(\frac{2\pi}{m_e T}\right)^{3/2} e^{E_I/T}$$

$$= \frac{2\zeta(3)}{\pi^2}\eta T^3\left(\frac{2\pi}{m_e T}\right)^{3/2} e^{E_I/T}, \quad \zeta(s) = \sum_{n=1}^{\infty}\frac{1}{n^s}$$

$$\boxed{\frac{1-X_e}{X_e^2} = \frac{2\zeta(3)}{\pi^2}\eta\left(\frac{2\pi T}{m_e}\right)^{3/2} e^{E_I/T}} \leftarrow \text{Saha equation}$$

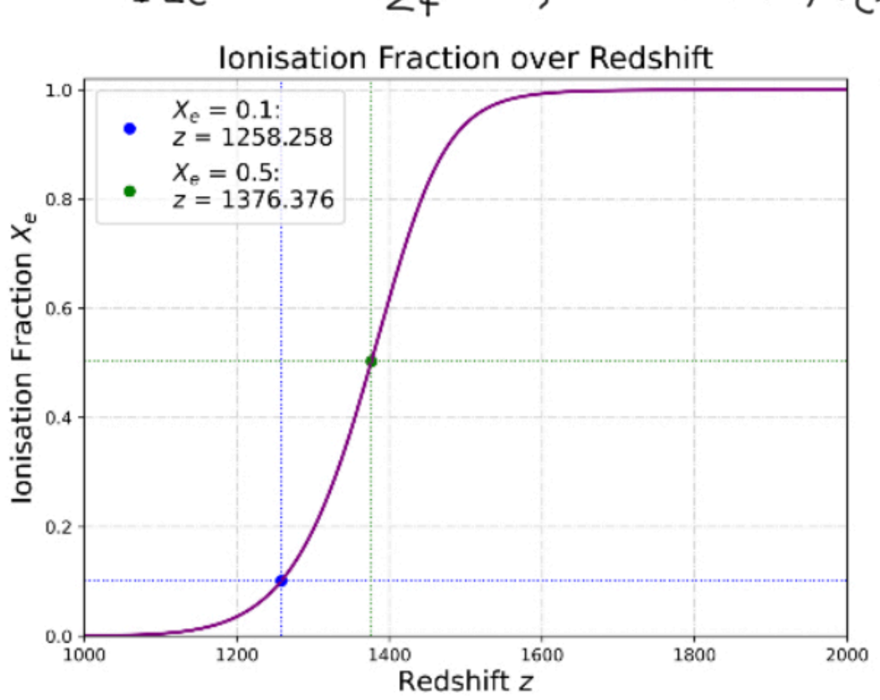
b.) Solve for X_e as function of z. Plot.

(Did not solve by hand, but rearranged for quadratic formula.)

$$X_e = \frac{-1+\sqrt{1+4f}}{2f}, \text{ where } f(T,\eta) = \frac{2\zeta(3)}{\pi^2}\eta\left(\frac{2\pi T}{m_e}\right)^{3/2} e^{E_I/T}$$

$$= \frac{2\zeta(3)}{\pi^2}\eta\left(\frac{2\pi T_0(1+z)}{m_e}\right)^{3/2} e^{E_I/T_0(1+z)}$$

$T_0 = 0.8\,\text{MeV}$



c.) From figure, find z at $X_e = 0.1$. How different from $z(X_e = 0.5)$?

See plot markers and labels:

- for $X_e = 0.1 \rightarrow z = 1258.258$
- for $X_e = 0.5 \rightarrow z = 1376.376$

Difference of over $z = 100$

d.) Calculate age of Universe at this epoch. Assume matter-dominated.

$$t = \frac{2}{3H}$$

$\downarrow$ Friedmann with $\Omega_m = 1 \rightarrow H = H_0\Omega_m a^{-3/2}$, $a = \frac{1}{1+z}$

$$t = \frac{2}{3}\frac{a^{+3/2}}{H_0\Omega_m}$$

When distance units cancel: $H_0 = 0.07\frac{1}{\text{Gyr}}$

$$= \frac{2}{3}\frac{1}{H_0\Omega_m(1+z)^{3/2}}$$

$$= \frac{2}{3}\,\frac{\text{Gyr}}{0.07}\,\frac{1}{1}\,\frac{1}{(1+1258.258)^{3/2}}$$

$$= 2.131\times 10^{-4}\ \text{Gyr}$$

$$\boxed{\begin{aligned} t &= 0.2131\ \text{Myr} \\ t &= 213{,}000\ \text{yr}\end{aligned}} \longrightarrow \text{Universe was very young at this ionisation fraction!}$$

e.) Using this ionisation fraction ($X_e = 0.1$), estimate z_{dec} where Thomson scattering rate = Hubble rate.

- Estimate Universe age at this z_{dec}. Find ionisation frac at this z_{dec}.

Interaction rate $\Gamma_\gamma \approx n_e\sigma_T$, where $\sigma_T \approx 2\times10^{-3}\,\text{MeV}^{-2}$; $n_e = n_b X_e$

At decoupling, $\Gamma_\gamma(T_{dec}) \approx H(T_{dec})$

Substitute for n_b to find $\Gamma_\gamma(T_{dec})$:

$$\Gamma_\gamma(T_{dec}) = n_b X_e(T_{dec})\sigma_T, \text{ where from 1a, } n_b = \eta n_\gamma = \eta\frac{2\zeta(3)}{\pi^2}T^3$$

$$= \frac{2\zeta(3)}{\pi^2}\eta\sigma_T X_e(T_{dec})T_{dec}^3$$

Solve for $H(T_{dec})$:

$$H(T_{dec}) = H_0\sqrt{\Omega_m}\left(\frac{T_{dec}}{T_0}\right)^{3/2}$$

$$c = \frac{3\times10^8\,\text{m}}{\text{sec}}\,\frac{\pi\times10^{16}\,\text{s}}{\text{Gyr}}\,\frac{1}{\text{m}\ 1.24\times10^{-12}\,\text{MeV}}$$

$$c = 7.60\times10^{36}\ \text{Gyr}^{-1}\,\text{MeV}^{-1}$$

$n\sigma v$

Set equal Γ_γ and H:

$$\frac{2\zeta(3)}{\pi^2}\eta\sigma_T X_e(T_{dec})T_{dec}^3 = H_0\sqrt{\Omega_m}\left(\frac{T_{dec}}{T_0}\right)^{3/2}$$

$$\underbrace{X_e(T_{dec})}_{0.1}\underbrace{T_{dec}}_{?}^{3/2} = \frac{\pi^2}{\underbrace{2\zeta(3)}_{1.2021}}\,\frac{H_0\sqrt{\Omega_m}}{\underbrace{\eta}_{6\times10^{-10}}\sigma_T T_0^{3/2}}$$

$H_0 = 0.07\frac{1}{\text{Gyr}}$; $\Omega_m = \cancel{0.3}\ 1$; $\sigma_T \approx 2\times10^{-3}\frac{1}{\text{MeV}^2}$; $2.73\,\text{K} = 2.3525\times10^{-10}\,\text{MeV}$

$$\frac{T^3}{T^{3/2}} = T^{3/2}$$

$$T_{dec} = \left[\frac{10}{}\,\frac{\pi^2}{2}\,\frac{}{1.2021}\,\frac{0.07}{\text{Gyr}}\,\overbrace{\frac{\text{Gyr}\ \text{MeV}}{7.60\times10^{36}}}^{c}\,\frac{\sqrt{1}}{\cancel{\sqrt{0.3}}}\,\frac{}{6\times10^{-10}}\,\frac{\text{MeV}^2}{2\times10^{-3}}\,\frac{}{(2.3525\times10^{-10})^{3/2}\,\text{MeV}^{3/2}}\right]^{2/3}$$

$$\boxed{T_{dec} = 2.579\times10^{-7}\ \text{MeV}}$$

- Estimate age:

$$T_{dec} = T_0(1+z_{dec})$$

$$z_{dec} = \frac{T_{dec}}{T_0} - 1$$

$$z_{dec} = \frac{2.480\times10^{-7}\,\text{MeV}}{2.3525\times10^{-10}\,\text{MeV}} - 1$$

$$\boxed{z_{dec} = 1095 \simeq 1100}$$

From my plot, $\boxed{X_e(z = 1095) = 0.004}$

$$t_{dec} = \frac{2}{3}\frac{a_{dec}^{+3/2}}{H_0\Omega_m}$$

$$= \frac{2}{3}\frac{1}{H_0\Omega_m(1+z_{dec})^{3/2}}$$

$$= \frac{2}{3}\,\frac{\text{Gyr}}{0.07}\,\frac{1}{1}\,\frac{1}{(1+1095)^{3/2}}$$

$$\boxed{t_{dec} = 262\ \text{Myr}}$$

$\boxed{\text{Universe would be 262 Myr with redshift 1095 and ionisation fraction 0.004.}}$

② "What If?" BBN

Neutron fraction defined as $X_n \equiv \frac{n_n}{n_n+n_p}$

a.) Derive equilibrium abundance $X_n = \frac{e^{-Q/T}}{1+e^{-Q/T}}$

where $Q = m_n - m_p = 1.3\,\text{MeV}$

From Baumann (3.126):

$$\left(\frac{n_n}{n_p}\right)_{eq} = \left(\frac{m_n}{m_p}\right)^{3/2} e^{-(m_n-m_p)/T}$$, Using equilibrium number densities as detailed in Baumann 3.73

Because m_n is very close to m_p, we disregard $\left(\frac{m_n}{m_p}\right)^{\frac{3}{2}}$:

$$\left(\frac{n_n}{n_p}\right)_{eq} = e^{-(m_n-m_p)/T}$$, where $Q = m_n - m_p$

$$\left(\frac{n_n}{n_p}\right)_{eq} = e^{-Q/T}$$

Putting this into the given X_n expression:

$$X_n \equiv \frac{n_n}{n_n+n_p}$$ → dividing by n_p in all terms

$$X_n = \frac{\frac{n_n}{n_p}}{\frac{n_n}{n_p}+\frac{n_p}{n_p}}$$

$$\boxed{X_n = \frac{e^{-Q/T}}{1+e^{-Q/T}}}$$ ← intended expression for neutron fraction X_n

b.) Estimate freeze-out abundance of neutrons X_n. Assume "freeze-out" temperature is 0.8 MeV

$$X_n = \frac{e^{-Q/T}}{1+e^{-Q/T}}$$, where $Q = 1.3\,\text{MeV}$, $T = 0.8\,\text{MeV}$

$$= \frac{e^{-1.3\,\text{MeV}/0.8\,\text{MeV}}}{1+e^{-1.3\,\text{MeV}/0.8\,\text{MeV}}}$$

$$= \frac{e^{-1.625}}{1+e^{-1.625}}$$

$$= \frac{0.19691}{1.19691}$$

$$\boxed{X_n = 0.1645}$$ ← freeze-out neutron abundance

c.) Calculate mass fraction of helium $Y_p = \frac{4n_{He}}{n_H}$.
Assume all neutrons from 4b are converted to helium-4.

No neutron decay, so all $n \to He$

* Important: The wording was unclear to me (and others), so my interpretation of 'assume all neutrons are converted' is that we are being told to not account for decay. I also think we're being told to use this Y_p expression and not derive it.
So that's what I solve here:

$$Y_p = \frac{4n_{He}}{n_H}$$, where $n_{He} = \frac{1}{2}n_n$ and $n_H \simeq n_p - n_n$

$$= \frac{4\cdot\frac{1}{2}n_n}{n_p - n_n}$$

$$= \frac{2\frac{n_n}{n_p}}{\frac{n_p}{n_p}-\frac{n_n}{n_p}}$$

$$= \frac{2X_n}{1-X_n}$$

$$= \frac{2(0.1645)}{1-0.1645}$$

$$\boxed{Y \simeq 0.39}$$

Mass fraction of Helium for freeze-out neutron abundance of $X_n = 0.1645$.

d.) If we maintained freeze-out temperature but doubled Q, how is He abundance impacted?

$T = 0.8\,\text{MeV}$

$Q = m_n - m_p = \cancel{1.3\,\text{MeV}}\ 2.6\,\text{MeV}$

$$X_n = \frac{e^{-Q/T}}{1+e^{-Q/T}}$$,

$$= \frac{e^{-2.6\,\text{MeV}/0.8\,\text{MeV}}}{1+e^{-26\,\text{MeV}/0.8\,\text{MeV}}}$$

$$= \frac{e^{-3.25}}{1+e^{-3.25}}$$

$$= \frac{0.03877}{1.03877}$$

$$X_n = 0.0373$$ ← freeze-out neutron abundance

Now use this to calculate helium mass fraction:

$$Y_p = \frac{4n_{He}}{n_H}$$

$$= \frac{4\cdot\frac{1}{2}n_n}{n_p-n_n}$$

$$= \frac{2\frac{n_n}{n_p}}{\frac{n_p}{n_p}-\frac{n_n}{n_p}}$$

$$= \frac{2X_n}{1-X_n}$$

$$= \frac{2(0.0373)}{1-0.0373}$$

$$\boxed{Y \simeq 0.078}$$

3. Freeze-In DM

Huterer 6.4

DM particle χ produced by $\sigma \to \chi\chi$ and $\delta n_\chi = -2\delta n_\sigma$ with $\Gamma \equiv \Gamma_{\sigma\to\chi\chi}$ (no n dependence; only on σ).

For small enough Γ, n_χ increases then flattens when n_σ becomes Boltzmann suppressed: $n_\sigma \propto e^{-m_\sigma/T} \ll 1$

a) FREEZE-OUT

Boltzmann for freezeout:

$$\frac{1}{a^3}\frac{d(na^3)}{dt} = -\langle\sigma v\rangle\left[n_1 n_2 - \left(\frac{n_1 n_2}{n_3 n_4}\right)_{eq} n_3 n_4\right] = -\langle\sigma v\rangle\left[n^2 - n_{eq}^2\right]$$

For freezeout Boltzmann eq: $\frac{dY}{dx} = -\frac{\lambda}{x^2}\left[Y^2 - Y_{eq}^2\right]$

FREEZE-IN

Boltzmann for freeze-in:

$$\frac{1}{a^3}\frac{d(na^3)}{dt} = 2\Gamma h(t) n_{\sigma,eq}(t), \text{ where } h(x) \simeq \frac{x}{x+2}, \quad x \equiv \frac{m_\sigma}{T}$$

Show for freeze-in Boltzmann: $\frac{dY}{dx} = \lambda_1 x h(x) Y_{eq}(x)$

Solution:

$$\Gamma_1 = n_2\langle\sigma v\rangle, \quad s \equiv \frac{S}{V} = \frac{\rho + P}{T} = \frac{2\pi^2}{45} g_{*s} T^3 \quad \text{(H 4.32)}$$

$$Y \equiv \frac{n}{T^3} \propto \frac{n}{s}, \quad x \equiv \frac{M}{T}, \quad \lambda \equiv \frac{2\pi^2}{45} g_{*s} \frac{m^3\langle\sigma v\rangle}{H(T=m)}$$

$$\frac{\lambda}{s} = \frac{m^3\langle\sigma v\rangle}{T^3 H(T=m)}$$

$$\langle\sigma v\rangle = \frac{\lambda T^3 H(T=m)}{s\, m^3}$$

$$\frac{dx}{dt} = Hx, \quad T \propto a^{-1}$$

$$\frac{1}{a^3}\frac{d(na^3)}{dt} = 2\Gamma h(t) n_{\sigma,eq}(t)$$

$$= 2 n_2\langle\sigma v\rangle \frac{t}{t+2} n_{\sigma,eq}(t)$$

$$= 2 n_2 \langle\sigma v\rangle \frac{\frac{m_\sigma}{T}}{\frac{m_\sigma}{T}+2} n_{\sigma,eq}(t)$$

$$T^3\frac{dY}{dt} = 2 n_2 \frac{\lambda T^3 H(T=m)}{s\, m^3} \frac{\frac{m_\sigma}{T}}{\frac{m_\sigma}{T}+2} n_{\sigma,eq}(t)$$

$$\boxed{\frac{dY}{dx} = \lambda_1 x h(x) Y_{eq}(x)}$$

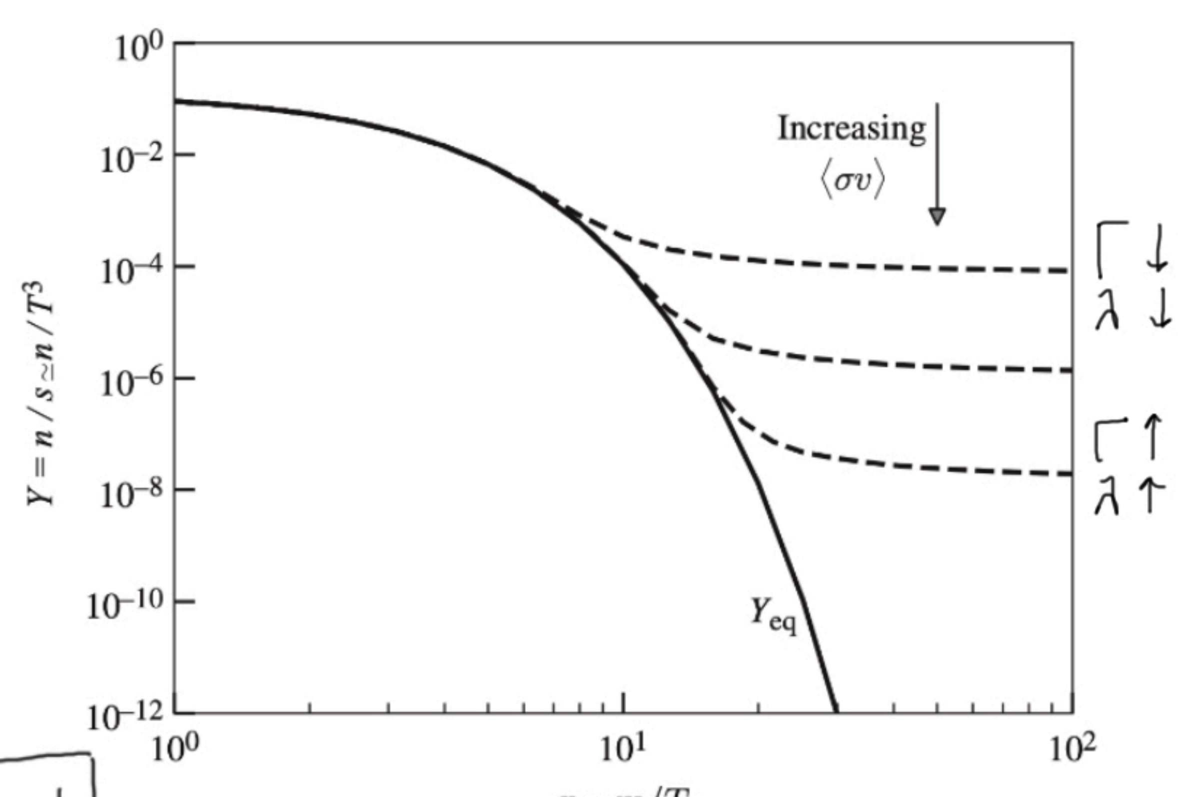


FIG 6.1

Freezeout of particle species. The y-axis is the comoving number-density-to-entropy ratio $Y \equiv n/s$, which is proportional to the comoving number density of particles (na^3). The x-axis is the mass of the particle divided by the temperature of the universe, $x = m/T$; basically, time flows to the right. The freezeout abundance depends on the annihilation cross-section of the particle multiplied by its velocity; the more efficient annihilation is (larger $\langle\sigma v\rangle$), the lower the final abundance. The three dashed curves correspond respectively to cases $\lambda = 10^5, 10^7, 10^9$; see text for details.

b.) Plot $Y(x)$

$$\frac{dY}{dx} = \lambda_1 x h(x) Y_{eq}(x), \quad \text{where } Y_{eq} \equiv \frac{n_{eq}}{T^3} = \frac{n_{eq} x^3}{M^3}$$

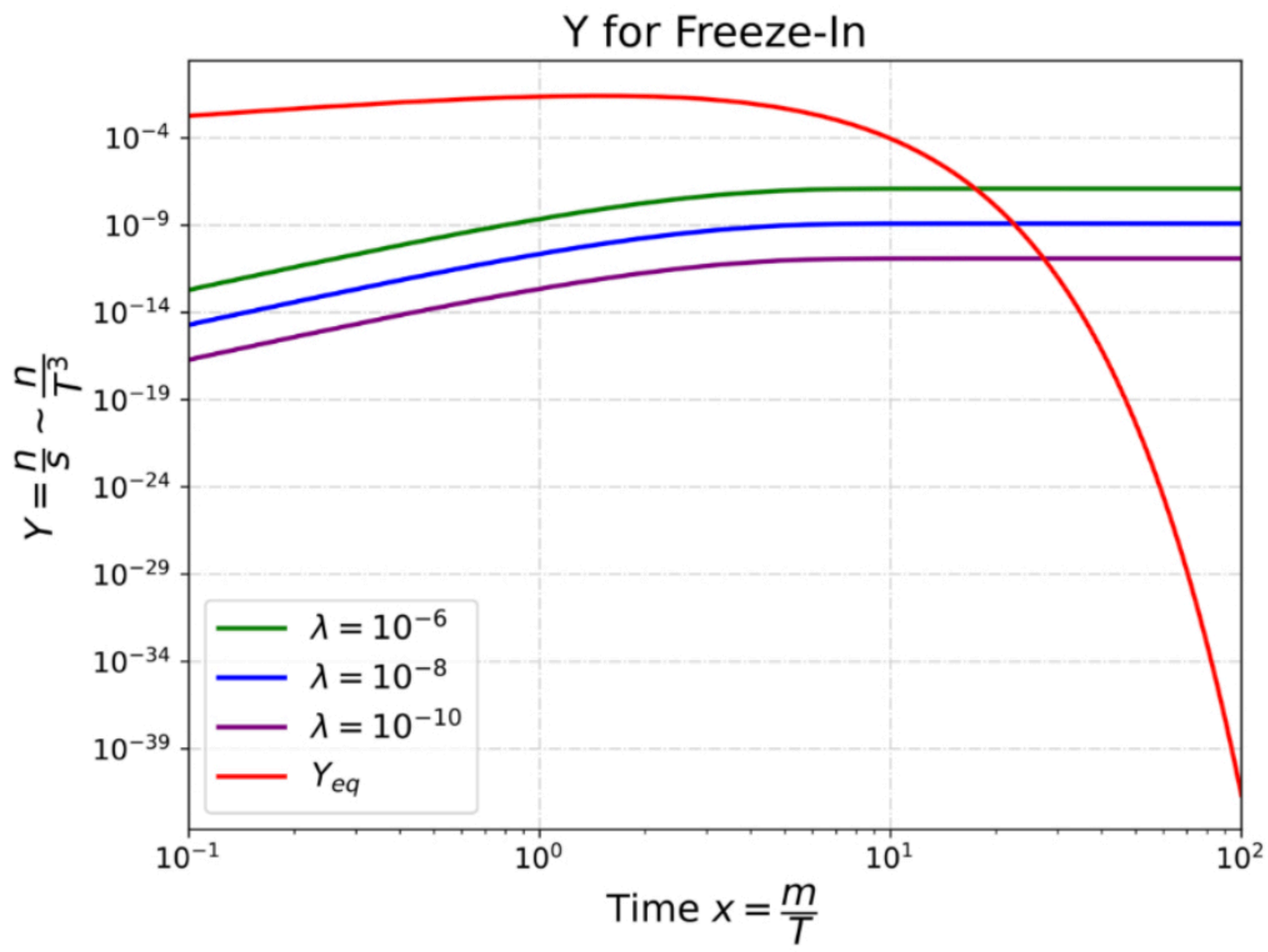
$$\frac{dY}{dx} = \lambda_1 x \left(\frac{x}{x+2}\right)\frac{n_{eq} x^3}{M^3}$$

$x \ll 1 \quad Y_{eq} \simeq 0.1$

$x \gg 1 \quad Y_{eq} \sim \left(\frac{x}{2\pi}\right)^{\frac{3}{2}} e^{-x}$

$$\int_{10^{-20}} dY = \lambda_1 \int_{0.01}^{100} \left(\frac{x^2}{x+2}\right)\left(\frac{x}{2\pi}\right)^{\frac{3}{2}} e^{-x} dx$$

I integrated and plotted this in my code file. Here's the result:

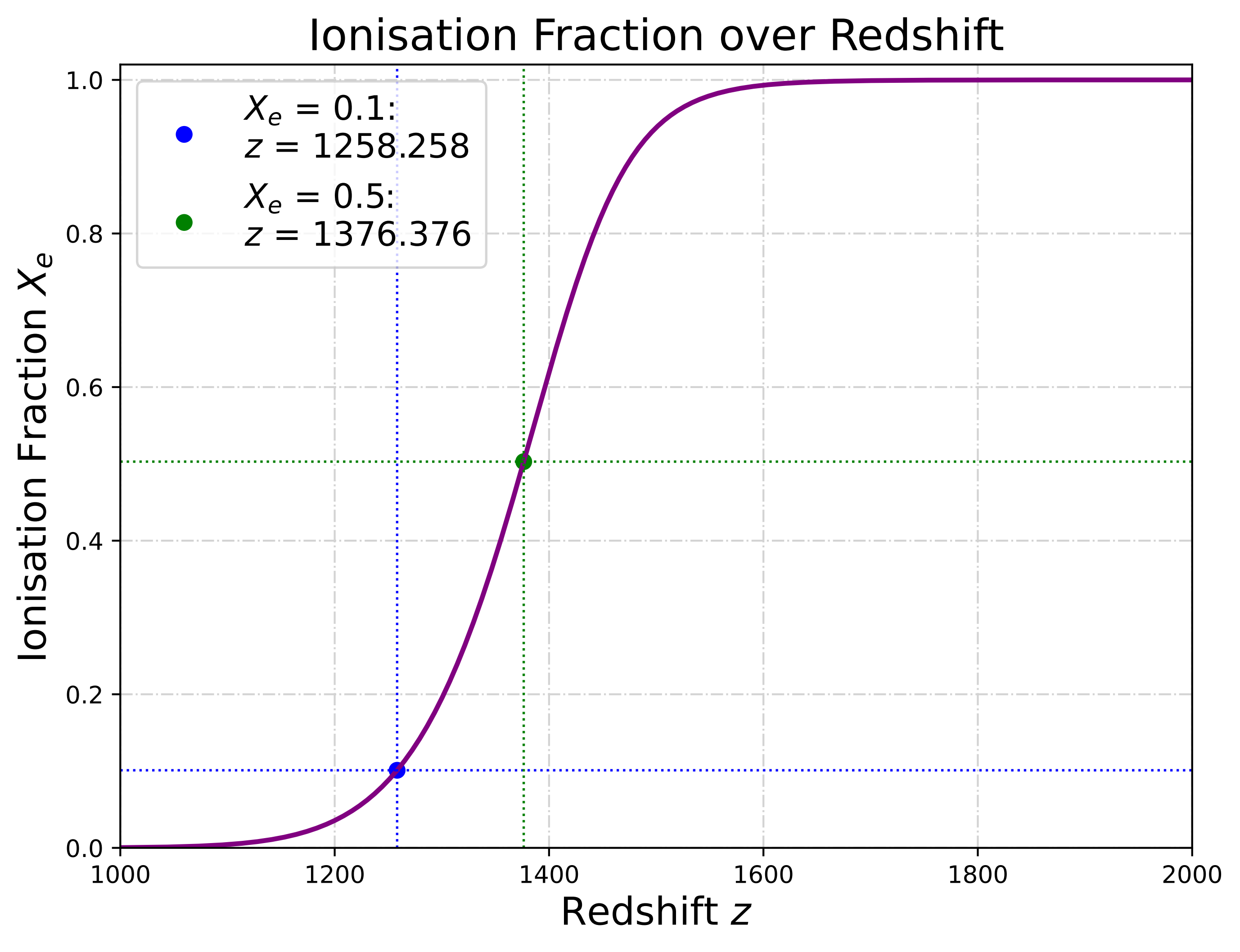


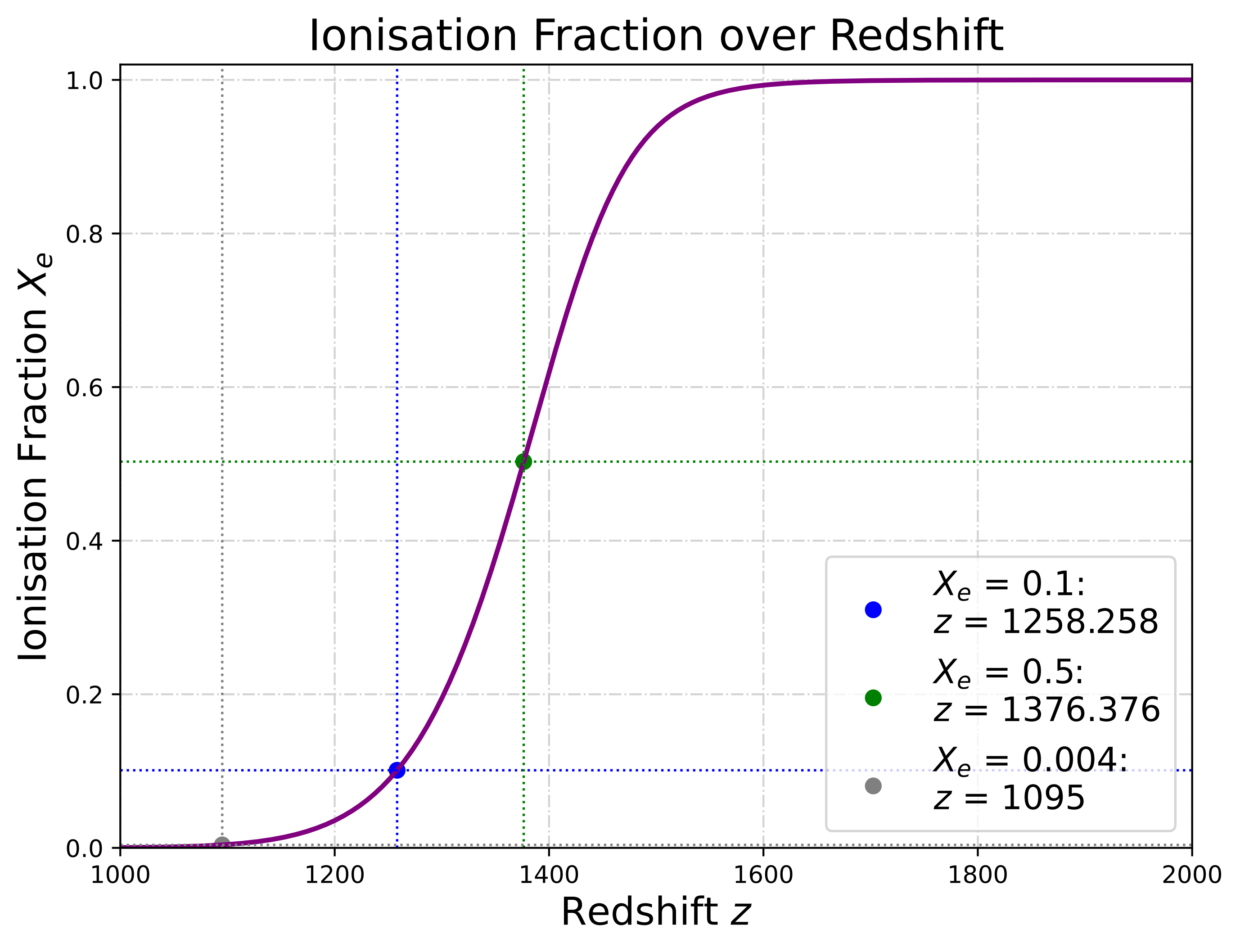
c.) Difference between freeze-out and freeze-in:

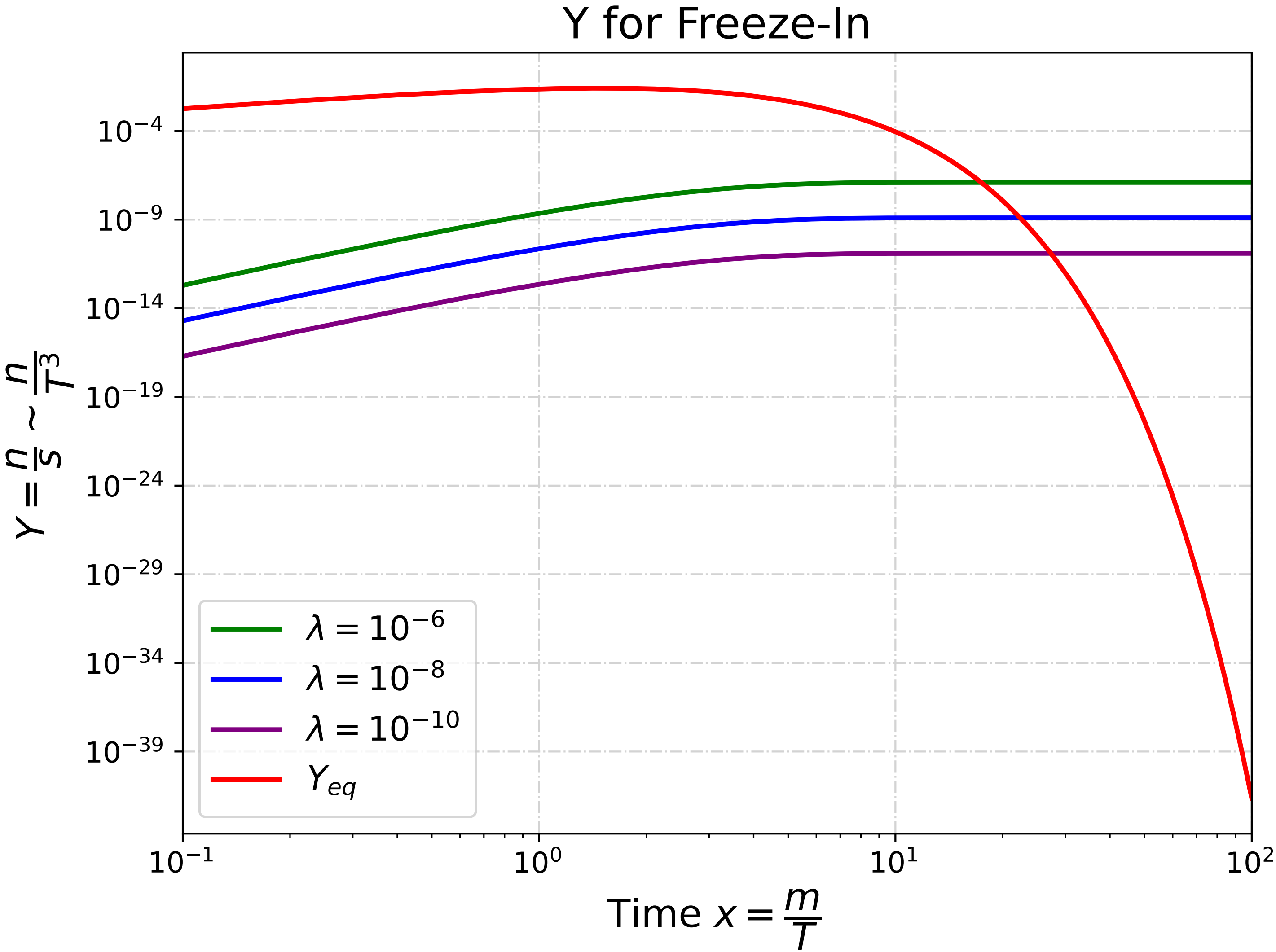
In terms of time dependence, we see that for freeze-out the Y_χ curves decrease before levelling out, whereas for freeze-in the Y_χ curves increase before levelling out.

Additionally, the Y_{eq} curve drops off much earlier in time for freeze-in than freeze-out.

Regarding the impact of the reaction rate Γ as this question asks, in Huterer 6.1 to the left for freeze-out, the greater reaction rates are lower on the plot, meaning that annihilation efficiency correlates with lower final abundances. The opposite is actually true of the freeze-in scenario, as our higher-λ curves (higher reaction rate Γ) are actually at higher Y_χ values.


Ionisation Fraction over Redshift
X_e = 0.1:
z = 1258.258
X_e = 0.5:
z = 1376.376
Ionisation Fraction X_e
Redshift z
1.0
0.8
0.6
0.4
0.2
0.0
1000
1200
1400
1600
1800
2000


Ionisation Fraction over Redshift
1.0
0.8
0.6
0.4
0.2
0.0
1000
1200
1400
1600
1800
2000
Redshift z
Ionisation Fraction X_e
X_e = 0.1:
z = 1258.258
X_e = 0.5:
z = 1376.376
X_e = 0.004:
z = 1095



Y for Freeze-In
10^{-4}
10^{-9}
10^{-14}
10^{-19}
10^{-24}
10^{-29}
10^{-34}
10^{-39}
$Y = \frac{n}{s} \sim \frac{n}{T^3}$
10^{-1}
10^{0}
10^{1}
10^{2}
Time $x = \frac{m}{T}$
$\lambda = 10^{-6}$
$\lambda = 10^{-8}$
$\lambda = 10^{-10}$
Y_{eq}

ASTR 600 - Cosmology

HW 4

Iver Warburton

October 2023

In [1]:
```
# Our lovely imports

import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import quad
from scipy.special import zeta
```

Problem 1

part ii.)

In [2]:
```
# Constants

# Masses
m_p = 938.272 # MeV/c^2
m_n = 940.6 # MeV/c^2
m_e = 0.511 # MeV/c^2
m_phot = 0
m_H = 938.8  #939.0 # MeV/c^2

# Number densities
#n_p =
#n_n =
#n_e =
#n_phot =

#n_H = n_p - n_n
#n_He = (1/2)*n_n
#n_B = n_p + n_H

# Defined
E_I = 0.0000136 # MeV, so 13.6 eV  #m_p + m_e - m_H
print(E_I)
eta = 6 * 10**-10  # ___  #n_B/n_gamma
```

```
1.36e-05
```

In [3]:
```
z = np.linspace(1000, 2000, 1000)

T_o = 2.3525 * 10**-10  #MeV  #2.73 K  ###0.8 MeV
T = T_o * (1 + z)
```

In [4]:
```
# Build the solution function
def build_Xe_solution(T, eta, E_I, m_e):
    f = ((2*zeta(3))/(np.pi**2)) * eta * (((2*np.pi*T)/m_e)**(3/2)) *
    X_e = (-1 + np.sqrt(1 + (4*f))) / (2*f)
    return X_e
```

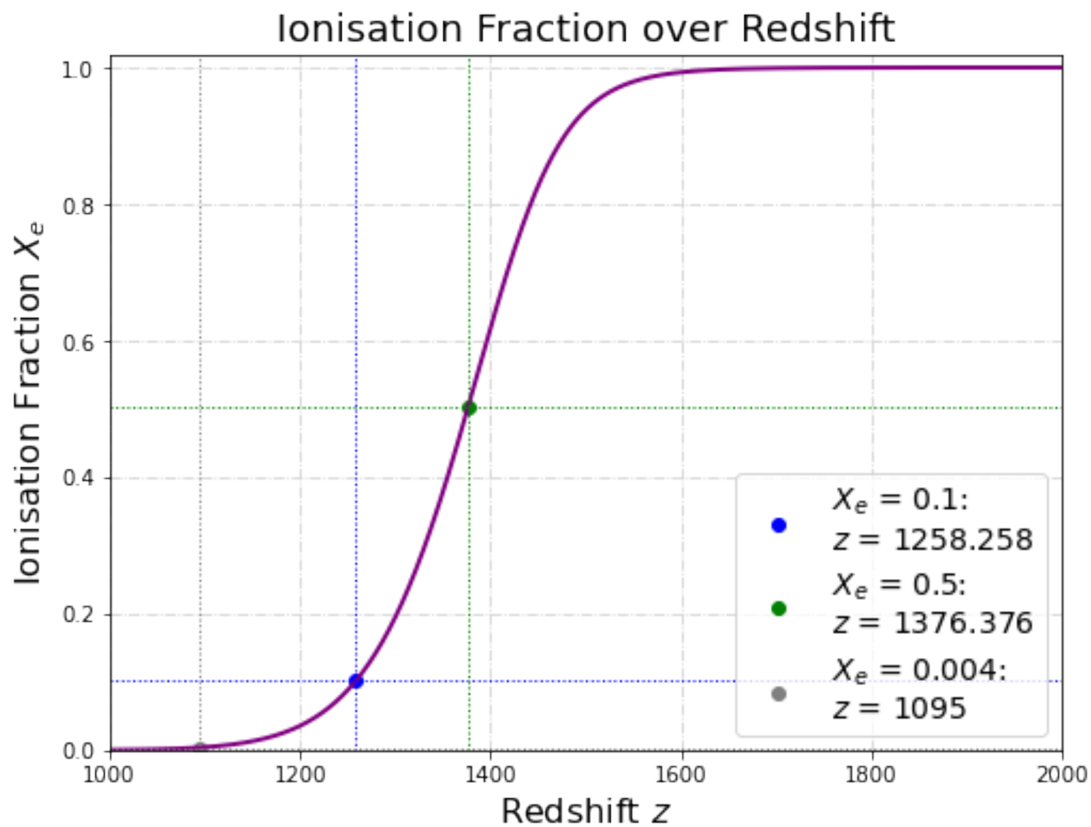
In [5]:
```
‡ Plot results
lef Xe_solution_plot(z, X_e):

   ## PLOT ##
   # Set figure
   fig, ax = plt.subplots(figsize=(8,6))

   # Plot
   plt.plot(z, X_e, ls='-', color='purple', lw=2)


   ## POINTS ##
   ## For X_e = 0.1  ##
   # Add point
   index_Xe01 = np.where(X_e >= 0.1)[0][0]
   Xe01 = X_e[index_Xe01]
   z_Xe01 = z[index_Xe01]
   print("z at X_e = 0.1:", z_Xe01.round(3))
   plt.scatter(z[index_Xe01], X_e[index_Xe01], color='blue', label='$X

   # Add pointer lines
   plt.axhline(Xe01, ls=':', c='blue', lw=1)
   plt.axvline(z_Xe01, ls=':', c='blue', lw=1)


   ## For X_e = 0.5 ##
   # Add point
   index_Xe05 = np.where(X_e >= 0.5)[0][0]
   Xe05 = X_e[index_Xe05]
   z_Xe05 = z[index_Xe05]
   print("z at X_e = 0.5:", z_Xe05.round(3))
   plt.scatter(z[index_Xe05], X_e[index_Xe05], color='green', label='$

   # Add pointer lines
   plt.axhline(Xe05, ls=':', c='green', lw=1)
   plt.axvline(z_Xe05, ls=':', c='green', lw=1)


   ## Part 1e ##
   ## For z=1095 ##
   # Add point
```

```python
index_z1095 = np.where(z >= 1095)[0][0]
z_1095 = z[index_z1095]
Xe_z1095 = X_e[index_z1095]
print("X_e at z = 1095:", Xe_z1095.round(3))
plt.scatter(z[index_z1095], X_e[index_z1095], color='grey', label='

# Add pointer lines
plt.axhline(Xe_z1095, ls=':', c='grey', lw=1)
plt.axvline(z_1095, ls=':', c='grey', lw=1)
## End of part 1e addition ##



## LABELLING ##
# Label axes
plt.xlabel('Redshift $z$', fontsize=16)
plt.ylabel('Ionisation Fraction $X_e$', fontsize=16)

# Increse axis numbering text size
ax.tick_params(axis='both', which='major', labelsize=10)
ax.tick_params(axis='both', which='minor', labelsize=10)

# Plot features
plt.xlim(1000, 2000)
plt.ylim(0, 1.02)
plt.legend(fontsize=14)
plt.grid(True, color='lightgrey', ls='-.')
plt.title('Ionisation Fraction over Redshift', fontsize=18)


## SAVE ##
# Save and show
plt.savefig("HW4Q1ePlot.pdf", format="pdf", bbox_inches="tight", ov
plt.show()
```

In [6]:
```
Xe_solution_plot(z, build_Xe_solution(T, eta, E_I, m_e))
```

```
z at X_e = 0.1: 1258.258
z at X_e = 0.5: 1376.376
X_e at z = 1095: 0.004
```

```
/var/folders/1w/ktxtfrr91bj5bztz50dqm0fr0000gn/T/ipykernel_12415/3203
391629.py:74: MatplotlibDeprecationWarning: savefig() got unexpected
keyword argument "overwrite" which is no longer supported as of 3.3 a
nd will become an error in 3.6
  plt.savefig("HW4Q1ePlot.pdf", format="pdf", bbox_inches="tight", ov
erwrite=True)
```




```
build_Xe_solution(T, eta, E_I, m_e)
```

In [7]:
```
zeta(3)
```

Out[7]: 1.2020569031595942

In [ ]:

In [ ]:

In [ ]:

Problem 3

part ii.)

Numerical integration

In [8]:
```
# Define function to build integrand
def integrand(x):
    # Integrand expression for comoving distance that was provided in
    dY = ((x**2)/(x+2)) * (x/(2*np.pi))**(3/2) * np.exp(-x)
    return dY
```

```
lambdas = [10**-6, 10**-8, 10**-10]
x = np.linspace(0.01, 100, 1000)
Y_numint_list = {}

for lambda_val in lambdas:
    # Calculate the integral for comoving distance numerically
    Y_numint = lambda_val * quad(integrand, 0.1, 100)[0]
    #Y_numint = lambda_val * Y_numint
    Y_numint_list.append(Y_numint)
    print(np.shape(Y_numint))
```

In [30]:
```
lambda1 = 10**-6
lambda2 = 10**-8
lambda3 = 10**-10

x = np.linspace(0.01, 100, 1000)
#Y_numint_list = {}


# Calculate the integral numerically
Y1 = [lambda1 * quad(integrand, 0, x_val)[0] for x_val in x]
Y2 = [lambda2 * quad(integrand, 0, x_val)[0] for x_val in x]
Y3 = [lambda3 * quad(integrand, 0, x_val)[0] for x_val in x]

Yeq = (x/(2*np.pi))**(3/2) * np.exp(-x)

#Y_numint_list.append(Y_numint)
#print(np.shape(Y1))
#print(Y1)
```

In [43]:
```
# Plot results
def Y_plot(x, Y1, Y2, Y3):

    ## PLOT ##
    # Set figure
    fig, ax = plt.subplots(figsize=(8,6))

    # Plot
    plt.loglog(x, Y1, ls='-', color='green', lw=2, label='$\lambda = 1
    plt.loglog(x, Y2, ls='-', color='blue', lw=2, label='$\lambda = 10
    plt.loglog(x, Y3, ls='-', color='purple', lw=2, label='$\lambda =
    plt.loglog(x, Yeq, ls='-', color='red', lw=2, label='$Y_{eq}$')
    #for Y in Y_list:
        #plt.plot(x, Y, ls='-', color='purple', lw=2)


    ## LABELLING ##
    # Label axes
    plt.xlabel('Time $x = \dfrac{m}{T}$', fontsize=16)
    plt.ylabel('$Y = \dfrac{n}{s} \sim \dfrac{n}{T^3}$', fontsize=16)

    # Increse axis numbering text size
    ax.tick_params(axis='both', which='major', labelsize=12)
    ax.tick_params(axis='both', which='minor', labelsize=12)

    # Plot features
    plt.xlim(0.1, 100)
    #plt.ylim(0, 1.02)
    plt.legend(fontsize=14)
    plt.grid(True, color='lightgrey', ls='-.')
    plt.title('Y for Freeze-In', fontsize=18)


    ## SAVE ##
    # Save and show
    plt.savefig("HW4Q3Plot.pdf", format="pdf", bbox_inches="tight", ov
    plt.show()
```

In [44]:
```
Y_plot(x, Y1, Y2, Y3)
```

```
/var/folders/1w/ktxtfrr91bj5bztz50dqm0fr0000gn/T/ipykernel_12415/629557688.py:36: MatplotlibDeprecationWarning: savefig() got unexpected keyword argument "overwrite" which is no longer supported as of 3.3 and will become an error in 3.6
  plt.savefig("HW4Q3Plot.pdf", format="pdf", bbox_inches="tight", overwrite=True)
```