# Understanding Physical Aging and its Mechanical Characterization

Luigi Casagrande

2024.12.12

# Overview

**PHYSICAL AGING**

First introduced by Kovacs in 1963 to describe a phenomenon that was later extensively studied by Struik in 1978

Process by which a material's properties gradually change over time as it moves toward thermodynamic equilibrium

**AGING SHIFT FACTOR**

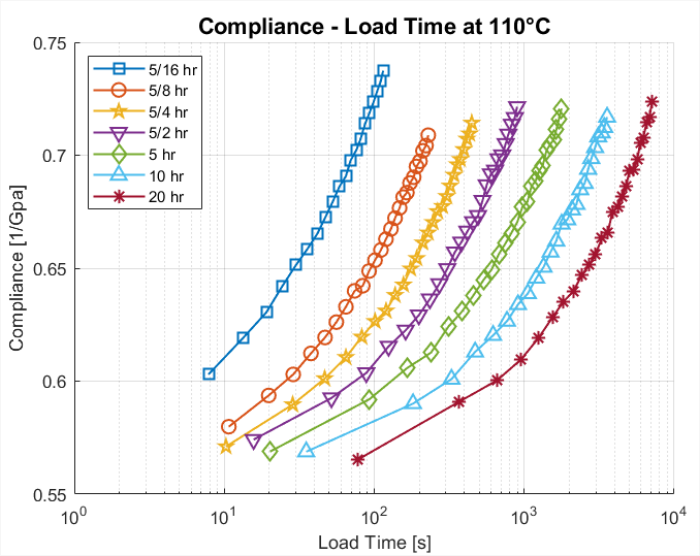
Aging is NOT random

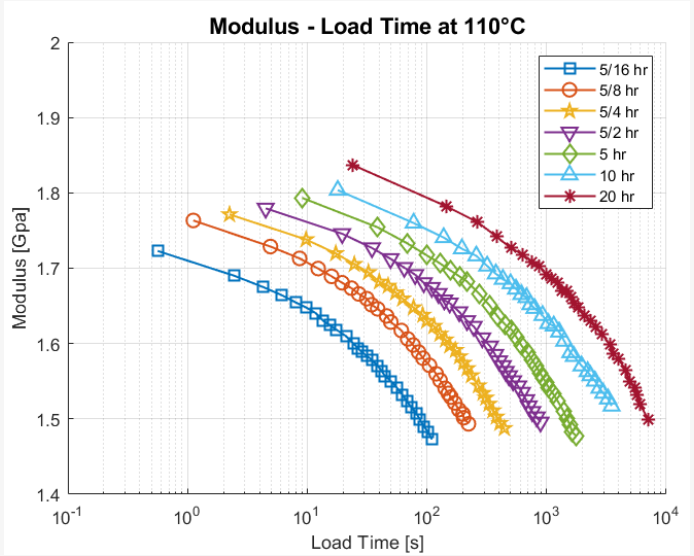

As a material ages, it may deform more slowly as its internal structure stabilizes

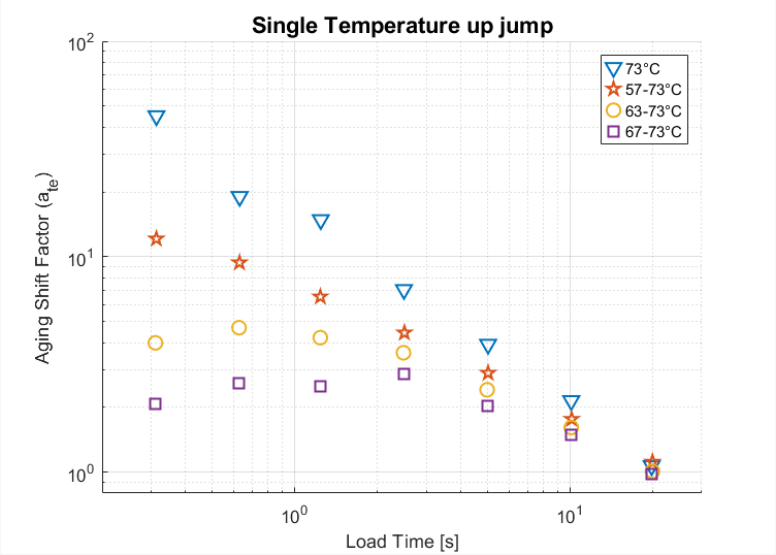
**"MATrix LABoratory"**

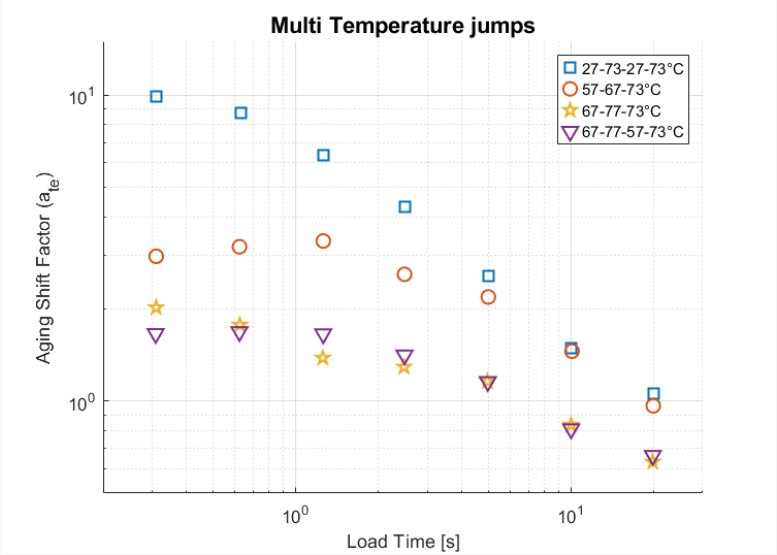
Programming language and numeric computing environment developed by MathWorks.

# Data Extraction

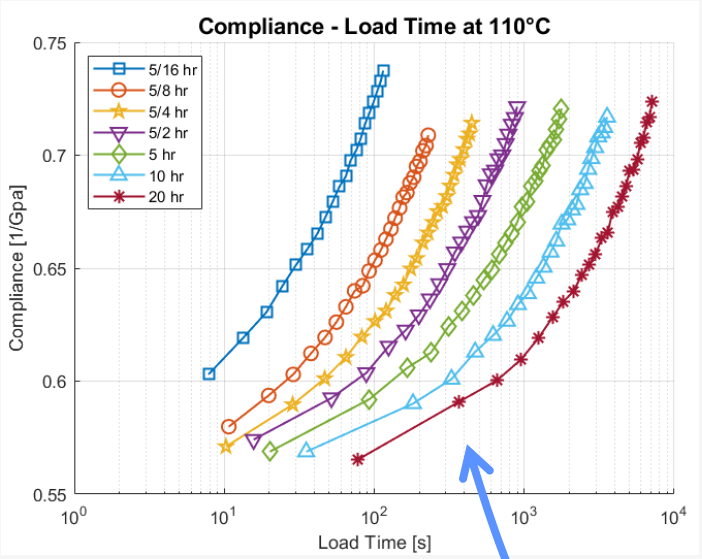


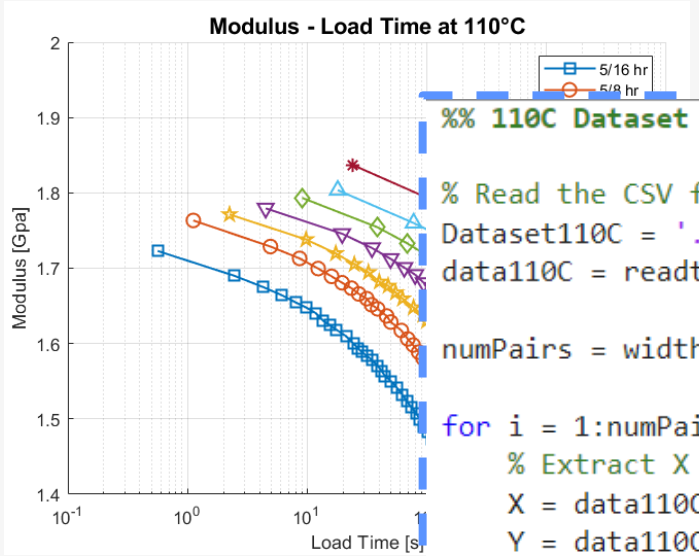



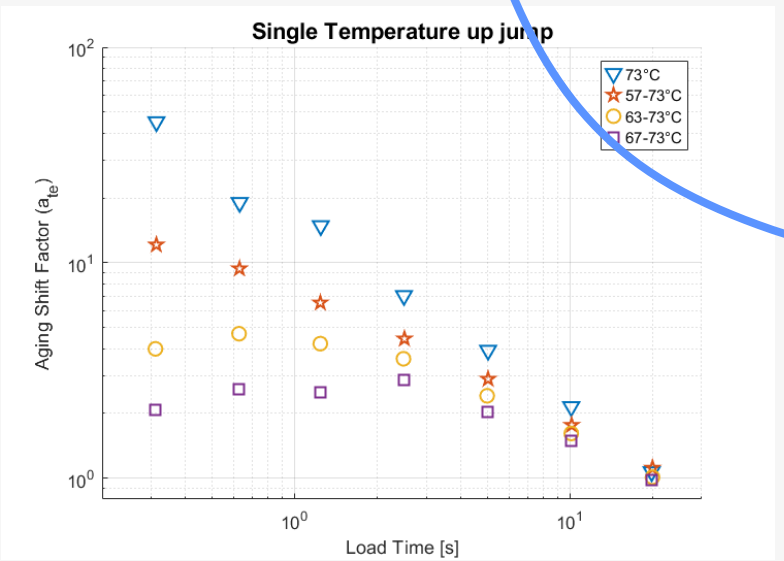



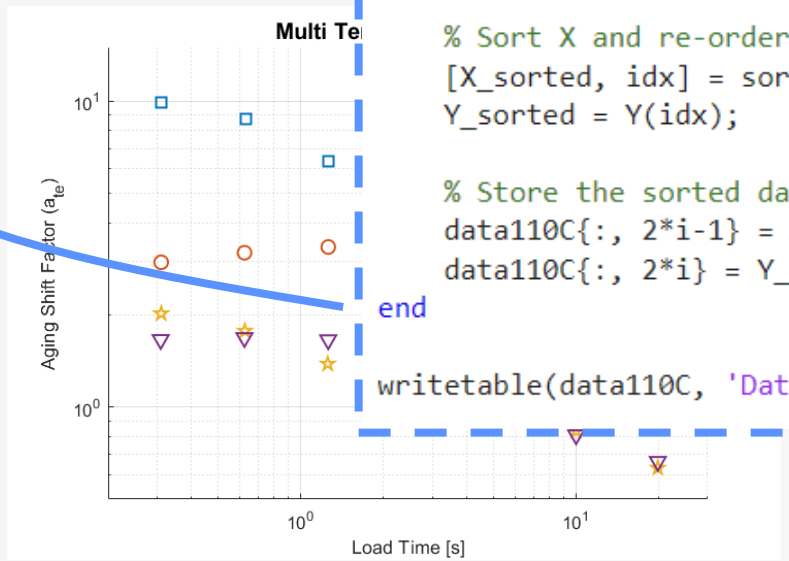



# Data Extraction




Extracted data into a CSV file

```
%% 110C Dataset

% Read the CSV file
Dataset110C = '../[03]-Raw_Data/Collected_data/110C_Dataset.csv';
data110C = readtable(Dataset110C,'ReadVariableNames',false);

numPairs = width(data110C) / 2;  % Number of (X,Y) pairs

for i = 1:numPairs
    % Extract X and Y columns for the current pair
    X = data110C{:, 2*i-1};  % X column (odd index)
    Y = data110C{:, 2*i};    % Y column (even index)

    % Sort X and re-order Y accordingly
    [X_sorted, idx] = sort(X);
    Y_sorted = Y(idx);

    % Store the sorted data back into the table
    data110C{:, 2*i-1} = X_sorted;
    data110C{:, 2*i} = Y_sorted;
end

writetable(data110C, 'DataSet/Data110C.xlsx', 'WriteVariableNames', false);
```

# 1| Characterization of Isothermal Physical Aging


Determine the horizontal shift needed to align the aging curves with a reference curve

```
% Reference curve (first one)
x_ref = x_data(:,1);
y_ref = y_data(:,1);

% Shift Factor Calculation
for j = 2:num_curves
    % Test curve
    x_test = x_data(:,j);
    y_test = y_data(:,j);

    % Definition of RMS error function
    rms_error_o = @(a_T_o) compute_rms_error_o(a_T_o, x_ref, x_test);
    rms_error_v = @(a_T_v) compute_rms_error_v(a_T_v, y_ref, y_test);

    % Initial guess for a_T
    initial_guess = 5;

    % Minimize the RMS error with custom options
    optimal_a_T_horizontal = fminsearch(rms_error_o, initial_guess, options);
    optimal_a_T_vertical = fminsearch(rms_error_v, initial_guess, options);
    a_T_o_store(:,j-1) = optimal_a_T_horizontal;
    a_T_v_store(:,j-1) = optimal_a_T_vertical;

    % Output
    disp(['Optimal a_T between reference curve 1 and curve ', num2str(j),' is: ', num2str
end
```

Reference curve as the one corresponding to the lowest aging time, specifically 5/16 hours

Aging shift factors for each test curve by minimizing the root-mean-square (RMS) error between the reference and each curve

Horizontal and vertical shifts

MATLAB's *fminsearch* function to minimize the error

# 1| Characterization of Isothermal Physical Aging

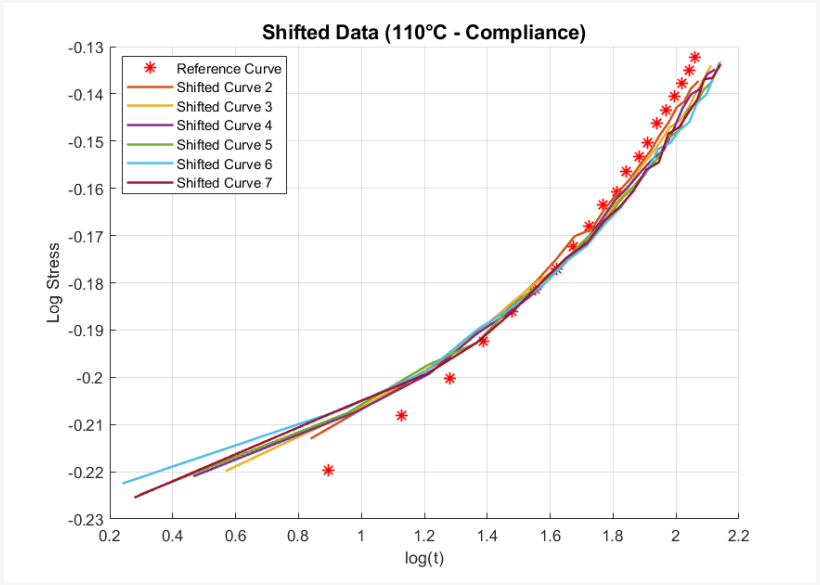

Determine the horizontal shift needed to align the aging curves with a reference curve

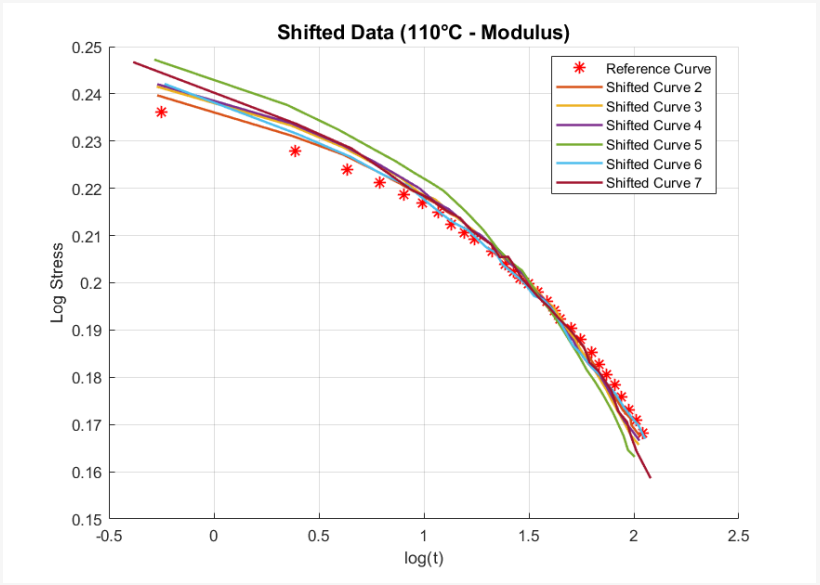
```
% Reference curve (first one)
x_ref = x_data(:,1);
y_ref = y_data(:,1);

% Shift Factor Calculation
for j = 2:num_curves
    % Test curve
    x_test = x_data(:,j);
    y_test = y_data(:,j);

    % Definition of RMS error function
    rms_error_o = @(a_T_o) compute_rms_error_o(a_T_o, x_ref, x_test);
    rms_error_v = @(a_T_v) compute_rms_error_v(a_T_v, y_ref, y_test);

    % Initial guess for a_T
    initial_guess = 5;

    % Minimize the RMS error with custom options
    optimal_a_T_horizontal = fminsearch(rms_error_o, initial_guess, options);
    optimal_a_T_vertical = fminsearch(rms_error_v, initial_guess, options);
    a_T_o_store(:,j-1) = optimal_a_T_horizontal;
    a_T_v_store(:,j-1) = optimal_a_T_vertical;

    % Output
    disp(['Optimal a_T between reference curve 1 and curve ', num2str(j),' is: ', num2str
end
```

```
%% Function definition
function error = compute_rms_error_v(a_T_v,y_ref,y_test)
    % Shift x_data by a_T
    shifted_y = y_ref + a_T_v;

    % Compute the RMS error
     error = sqrt(mean((shifted_y - y_test).^2));
end
function error = compute_rms_error_o(a_T_o,x_ref,x_test)
    % Shift x_data by a_T
    shifted_x = x_ref + a_T_o;

    % Compute the RMS error
    error = sqrt(mean((shifted_x - x_test).^2));
end
```

$$\mathrm{RMS}\,(a_T) = \sqrt{\frac{1}{n}\cdot\sum_{i=1}^{n}\left[x_{\mathrm{ref}}(i) - x_{\mathrm{shifted},j}(i)\right]^2}$$

# 1| Characterization of Isothermal Physical Aging

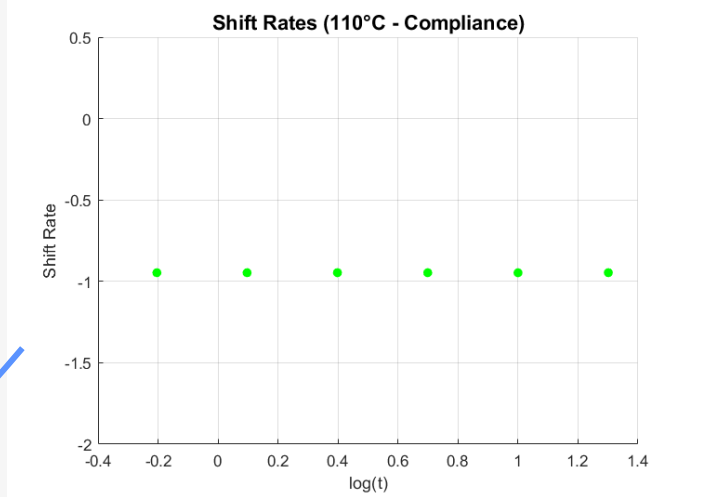


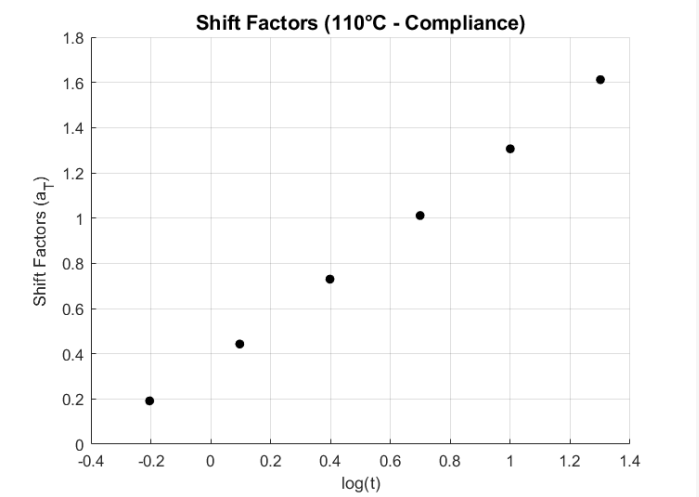




Shifted curve to validate the results obtained

log-log plot

Shift factor and shift rates plots for the 110°C creep compliance




Isothermal

# 1| Characterization of Isothermal Physical Aging

**Table 1.** Aging shift factors and aging shift rate along the $x$-axis for the six datasets analyzed. The curves have been renamed, for simplicity, according to the temperature they refer to, with "C" indicating the creep compliance curve and "M" indicating the stress relaxation modulus.

| | Shift Factor | | | | | | Shift Rate |
|---|---|---|---|---|---|---|---|
| **Plot** | **Curves 1-2** | **Curves 1-3** | **Curves 1-4** | **Curves 1-5** | **Curves 1-6** | **Curves 1-7** | **along $x$** |
| **110C** | 0.1914 | 0.4429 | 0.7299 | 1.0111 | 1.3061 | 1.6119 | -0.9467 |
| **110M** | 0.3185 | 0.6229 | 0.9227 | 1.2418 | 1.4899 | 1.7681 | -0.9651 |
| **120C** | 0.1703 | 0.4385 | 0.7239 | 1.0155 | 1.3109 | 1.6150 | -0.9617 |
| **120M** | 0.3210 | 0.6102 | 0.9187 | 1.1896 | 1.5174 | 1.8102 | -0.9912 |
| **130C** | 0.1371 | 0.3052 | 0.5839 | 0.8471 | 1.1337 | 1.4436 | -0.8809 |
| **130M** | 0.3013 | 0.5869 | 0.9206 | 1.1969 | 1.5039 | 1.8007 | -0.9989 |

# 2| Relationship between Creep and Stress Relaxation


Calculate the theoretical creep compliance D(t) from experimentally measured stress relaxation modulus E(t)

```
%% Fit the data
for j = 1:num_curves
    x_cf = x_in(:,j);
    y_cf = y_in(:,j);
    regression = fit(x_cf,y_cf,'poly2');
    coeffs(:,j) = [regression.p1,regression.p2,regression.p3];
end

% Laplace transformation
for k = 1:num_curves
    syms x s

    % Define the creep compliance polynomial function
    creep = coeffs(1,k)*x.^2+coeffs(2,k)*x + coeffs(3,k);

    % Laplace transform of creep
    E_laplace = laplace(creep,x,s);

    % Inverse Laplace relationship
    D_laplace = (1/((s.^2)*E_laplace));
    D_t_sym = ilaplace(D_laplace,s,x);

    % Substitute x_in values into D(t) to get the creep compliance over time
    D_t_vals = double(subs(D_t_sym,x,x_in(:,k)));
    D_t(:,k) = D_t_vals;
end

% Reference creep data for comparison
x_ref = x_data_c(:,1);
y_ref = y_data_c(:,1);
```

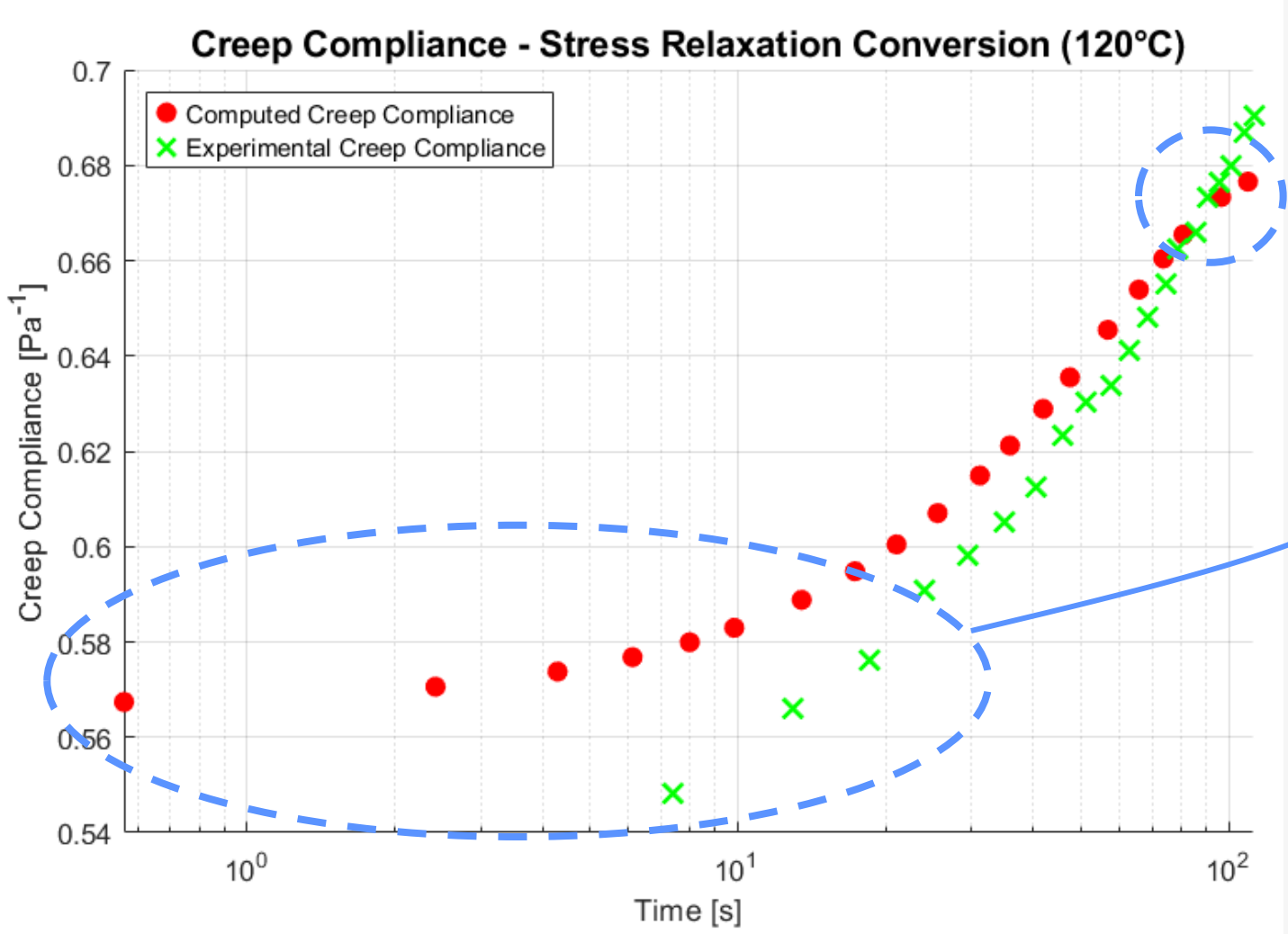
Fitted the stress relaxation modulus data for each curve to a quadratic polynomial

Laplace transform relationship between modulus and compliance

$$D(s) = \frac{1}{s^2 \cdot E(s)}$$

Comparison it with the experimental creep compliance data

# 2| Relationship between Creep and Stress Relaxation




The calculated compliance began to flatten out at both higher and lower times

1. **Finite Time Effects**
2. **Viscoelastic and Aging Effects**
3. **Mathematical Approximation**

# 3| Long-term Nonisothermal Physical Aging

## EFFECTIVE TIME THEORY

**Effective Time (λ):**
Accounts for the cumulative effects of aging

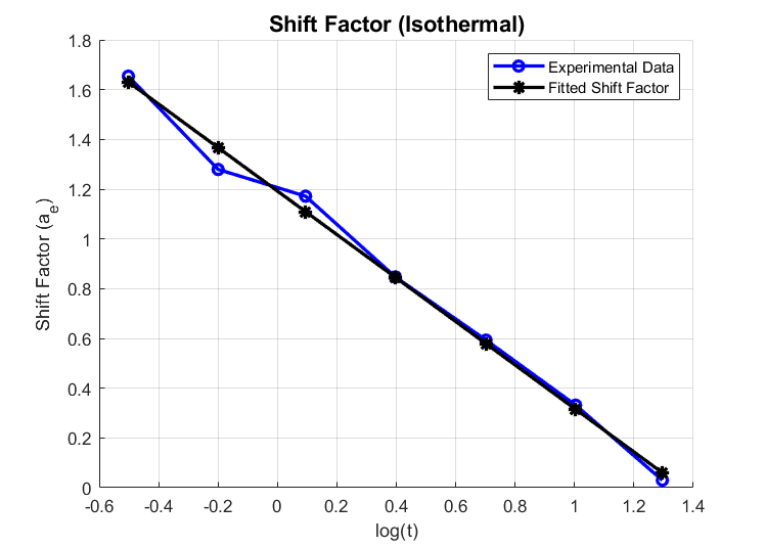
The theory works by transforming real time into "effective time", which reflects the material's evolving state as it undergoes aging.

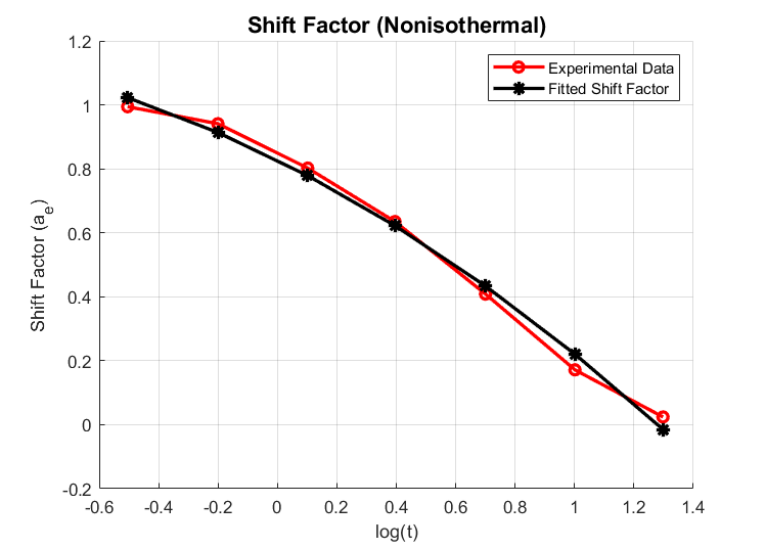
$$\mu = -\frac{\partial \log a_e(t)}{\partial \log t} \rightarrow a_e(t) = \left(\frac{t_0}{t_0 + t}\right)^{\mu}$$

$$\lambda(t) = \int_0^t a_e(x)\,dx \rightarrow D(t) = D_0 \cdot \exp\left[\left(\frac{\lambda(t)}{\tau}\right)^{\beta}\right]$$


Predict the 20-hour long-term creep response using aging shift factors and effective time theory under isothermal and nonisothermal conditions.

# 3| Long-term Nonisothermal Physical Aging



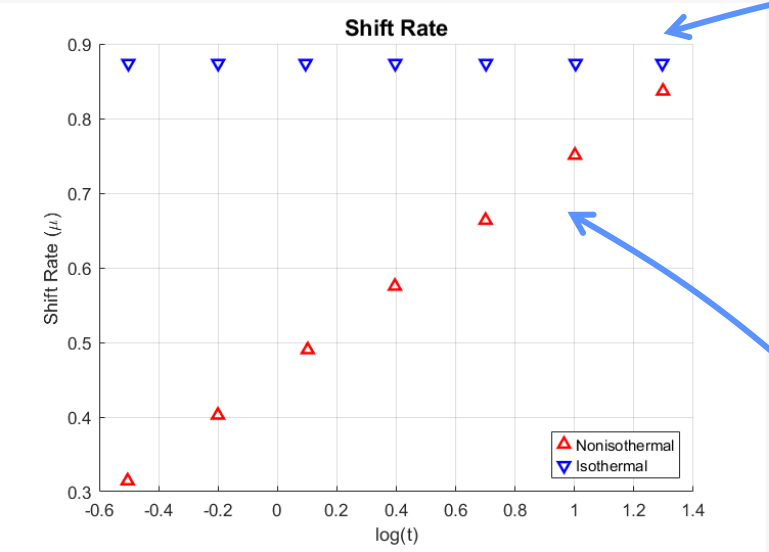

$$y = p_1 x + p_2$$

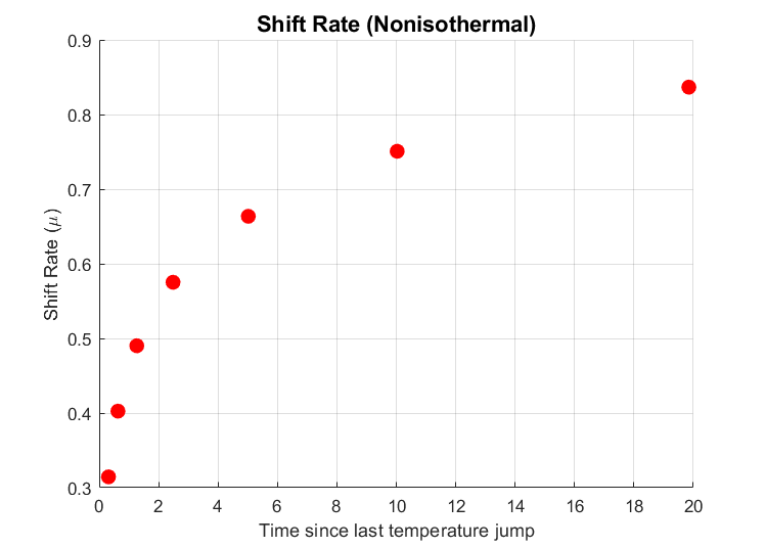
```
%% Regression for Isothermal Shift Factor/Rate
fitting_iso = fit(x_data_iso,y_data_iso,'poly1');
coeffs_iso = [fitting_iso.p1 fitting_iso.p2];
a_t_iso = coeffs_iso(1,1).*(x_data_iso) + coeffs_iso(1,2);

shift_rate_iso = - coeffs_iso(1,1);
a_u_iso_int = @(x) ((t0)./(t0 + x)).^(shift_rate_iso);

for i = 1:length(t_int)
    lambda_iso(i,1) = integral(a_u_iso_int,0,t_int(i,1));
    D_t_iso(i,1) = D_0 * exp((lambda_iso(i,1)./tau).^beta);
end
```

```
%% Regression for Nonisothermal Shift Factor/Rate
fitting_noniso = fit(x_data_noniso,y_data_noniso,'poly2');
coeffs_noniso = [fitting_noniso.p1,fitting_noniso.p2,fitting_noniso.p3];
a_t_noniso = (coeffs_noniso(1,1)*(x_data_noniso).^2 + ...
    coeffs_noniso(1,2)*x_data_noniso + coeffs_noniso(1,3));

shift_rate_noniso = - (2.*coeffs_noniso(1,1).*x_data_noniso + coeffs_noniso(1,2));
a_u_noniso = @(x) ((t0)./(t0+x)).^(-2.*coeffs_noniso(1,1).*x - coeffs_noniso(1,2));

for j = 1:length(t_int)
    lambda_noniso(j,1) = integral(a_u_noniso,0,t_int(j,1));
    D_t_noniso(j,1) = D_0 * exp((lambda_noniso(j,1)./tau).^beta);
end
```

$$y = p_1 x^2 + p_2 x + p_3$$

# 3| Long-term Nonisothermal Physical Aging



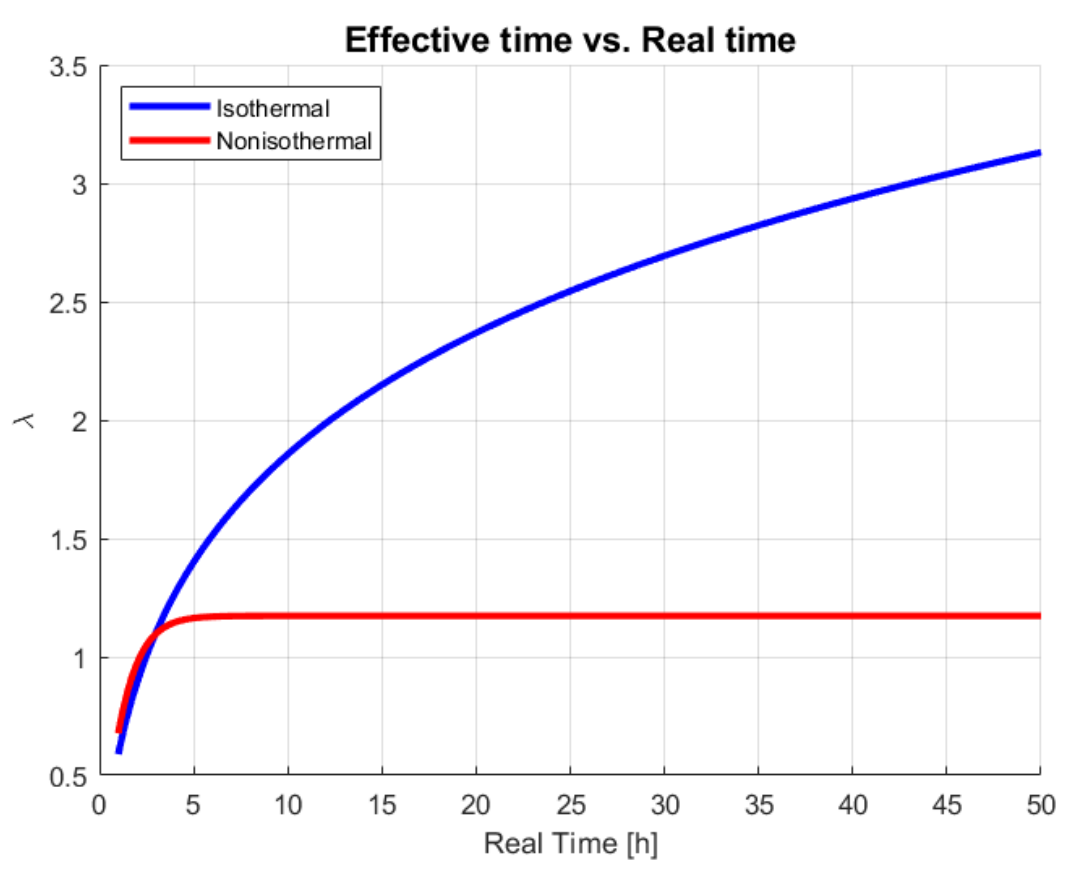




```
%% Regression for Isothermal Shift Factor/Rate
fitting_iso = fit(x_data_iso,y_data_iso,'poly1');
coeffs_iso = [fitting_iso.p1 fitting_iso.p2];
a_t_iso = coeffs_iso(1,1).*(x_data_iso) + coeffs_iso(1,2);

shift_rate_iso = - coeffs_iso(1,1);
a_u_iso_int = @(x) ((t0)./(t0 + x)).^(shift_rate_iso);

for i = 1:length(t_int)
    lambda_iso(i,1) = integral(a_u_iso_int,0,t_int(i,1));
    D_t_iso(i,1) = D_0 * exp((lambda_iso(i,1)./tau).^beta);
end
```

```
%% Regression for Nonisothermal Shift Factor/Rate
fitting_noniso = fit(x_data_noniso,y_data_noniso,'poly2');
coeffs_noniso = [fitting_noniso.p1,fitting_noniso.p2,fitting_noniso.p3];
a_t_noniso = (coeffs_noniso(1,1)*(x_data_noniso).^2 + ...
    coeffs_noniso(1,2)*x_data_noniso + coeffs_noniso(1,3));

shift_rate_noniso = - (2.*coeffs_noniso(1,1).*x_data_noniso + coeffs_noniso(1,2));
a_u_noniso = @(x) ((t0)./(t0+x)).^(-2.*coeffs_noniso(1,1).*x - coeffs_noniso(1,2));

for j = 1:length(t_int)
    lambda_noniso(j,1) = integral(a_u_noniso,0,t_int(j,1));
    D_t_noniso(j,1) = D_0 * exp((lambda_noniso(j,1)./tau).^beta);
end
```

$$\mu = -\frac{\partial \log a_e(t)}{\partial \log t}$$

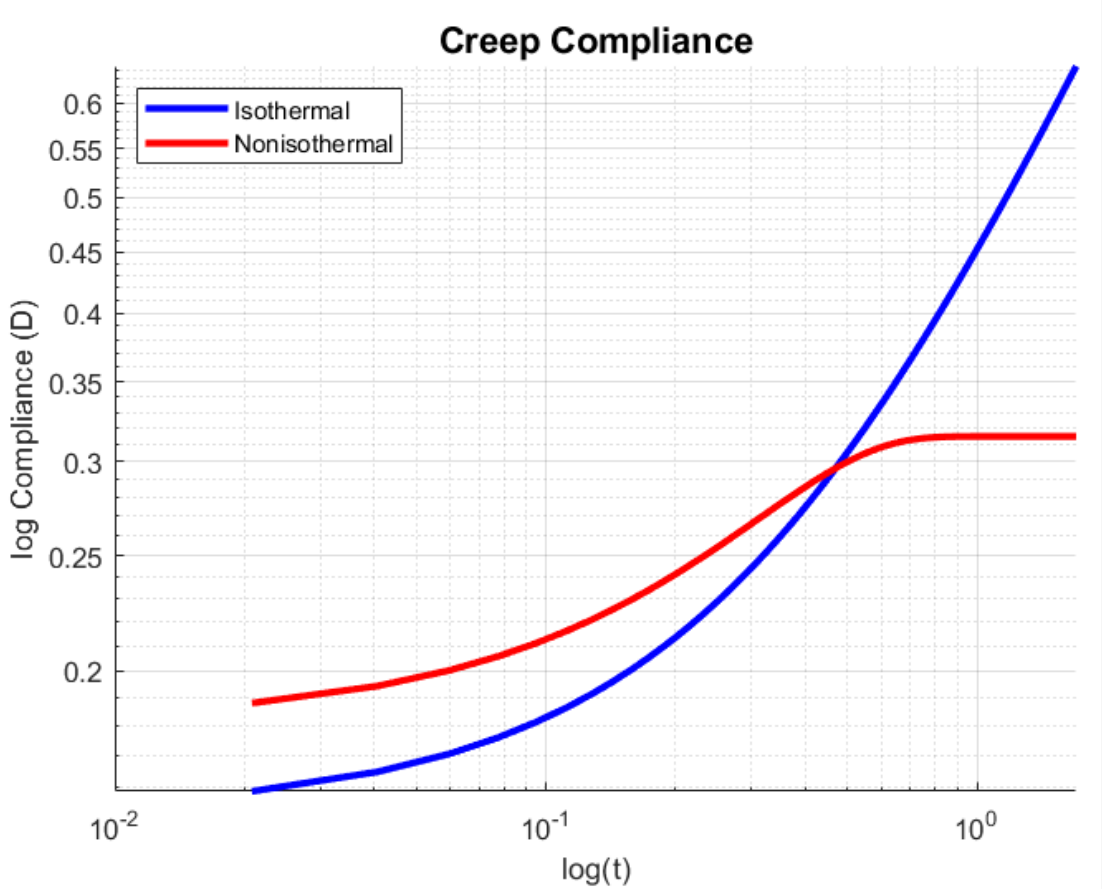
# 3| Long-term Nonisothermal Physical Aging




```
%% Regression for Isothermal Shift Factor/Rate
fitting_iso = fit(x_data_iso,y_data_iso,'poly1');
coeffs_iso = [fitting_iso.p1 fitting_iso.p2];
a_t_iso = coeffs_iso(1,1).*(x_data_iso) + coeffs_iso(1,2);

shift_rate_iso = - coeffs_iso(1,1);
a_u_iso_int = @(x) ((t0)./(t0 + x)).^(shift_rate_iso);

for i = 1:length(t_int)
    lambda_iso(i,1) = integral(a_u_iso_int,0,t_int(i,1));
    D_t_iso(i,1) = D_0 * exp((lambda_iso(i,1)./tau).^beta);
end
```

```
%% Regression for Nonisothermal Shift Factor/Rate
fitting_noniso = fit(x_data_noniso,y_data_noniso,'poly2');
coeffs_noniso = [fitting_noniso.p1,fitting_noniso.p2,fitting_noniso.p3];
a_t_noniso = (coeffs_noniso(1,1)*(x_data_noniso).^2 + ...
    coeffs_noniso(1,2)*x_data_noniso + coeffs_noniso(1,3));

shift_rate_noniso = - (2.*coeffs_noniso(1,1).*x_data_noniso + coeffs_noniso(1,2));
a_u_noniso = @(x) ((t0)./(t0+x)).^(-2.*coeffs_noniso(1,1).*x - coeffs_noniso(1,2));

for j = 1:length(t_int)
    lambda_noniso(j,1) = integral(a_u_noniso,0,t_int(j,1));
    D_t_noniso(j,1) = D_0 * exp((lambda_noniso(j,1)./tau).^beta);
end
```

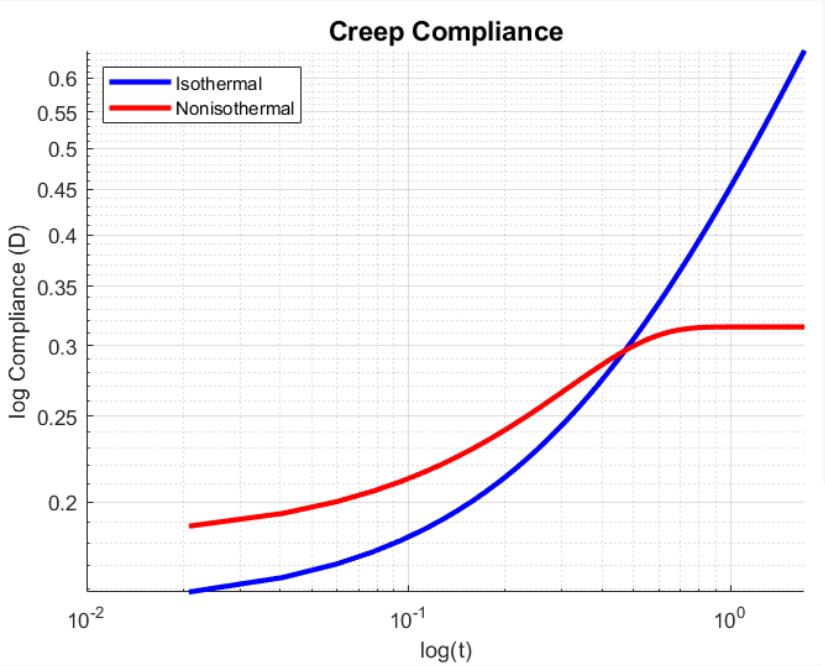
# 3| Long-term Nonisothermal Physical Aging



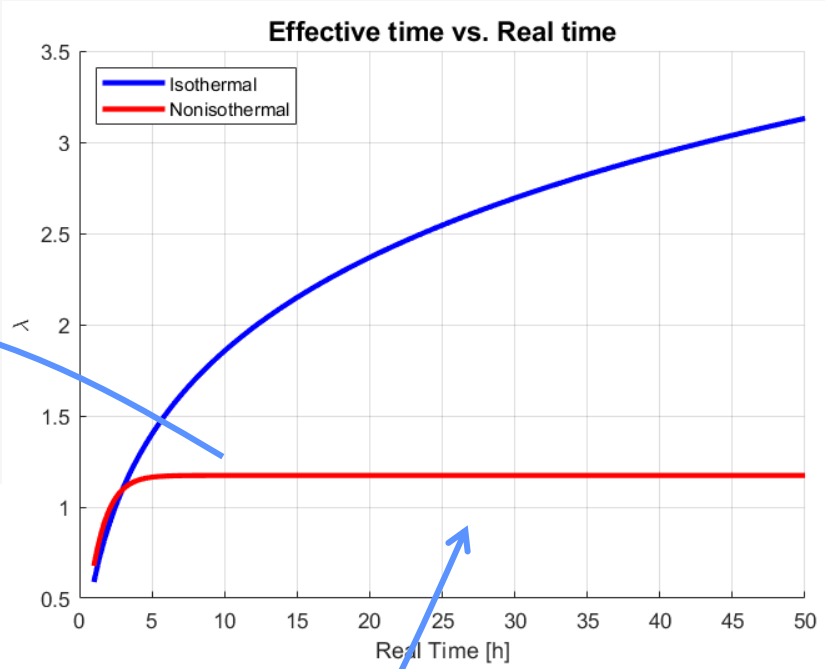

```
%% Regression for Isothermal Shift Factor/Rate
fitting_iso = fit(x_data_iso,y_data_iso,'poly1');
coeffs_iso = [fitting_iso.p1 fitting_iso.p2];
a_t_iso = coeffs_iso(1,1).*(x_data_iso) + coeffs_iso(1,2);

shift_rate_iso = - coeffs_iso(1,1);
a_u_iso_int = @(x) ((t0)./(t0 + x)).^(shift_rate_iso);

for i = 1:length(t_int)
    lambda_iso(i,1) = integral(a_u_iso_int,0,t_int(i,1));
    D_t_iso(i,1) = D_0 * exp((lambda_iso(i,1)./tau).^beta);
end
```

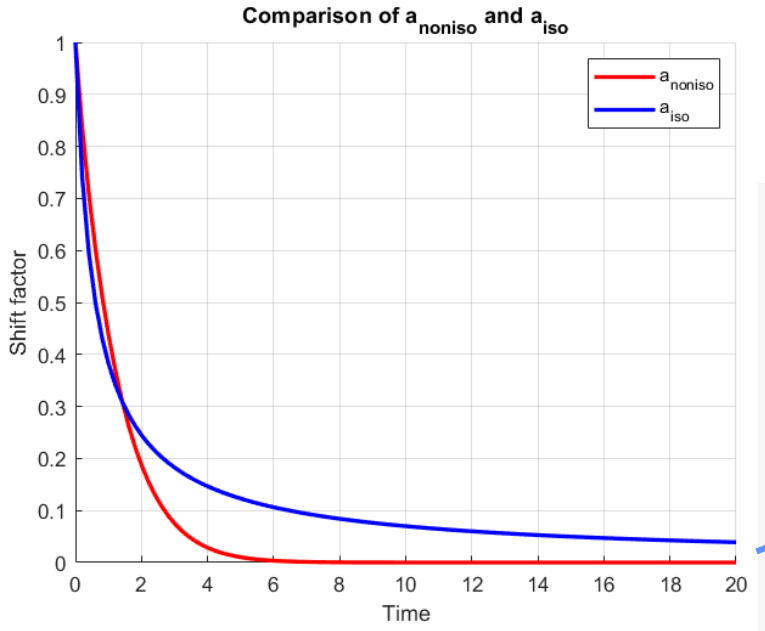
```
%% Regression for Nonisothermal Shift Factor/Rate
fitting_noniso = fit(x_data_noniso,y_data_noniso,'poly2');
coeffs_noniso = [fitting_noniso.p1,fitting_noniso.p2,fitting_noniso.p3];
a_t_noniso = (coeffs_noniso(1,1)*(x_data_noniso).^2 + ...
    coeffs_noniso(1,2)*x_data_noniso + coeffs_noniso(1,3));

shift_rate_noniso = - (2.*coeffs_noniso(1,1).*x_data_noniso + coeffs_noniso(1,2));
a_u_noniso = @(x) ((t0)./(t0+x)).^(-2.*coeffs_noniso(1,1).*x - coeffs_noniso(1,2));

for j = 1:length(t_int)
    lambda_noniso(j,1) = integral(a_u_noniso,0,t_int(j,1));
    D_t_noniso(j,1) = D_0 * exp((lambda_noniso(j,1)./tau).^beta);
end
```

# 3| Long-term Nonisothermal Physical Aging




$$D(t) = D_0 \cdot \exp\left[\left(\frac{\lambda(t)}{\tau}\right)^{\beta}\right]$$




$$a_{\text{noniso}} = \left(\frac{t_0}{t_0 + x}\right)^{-(2p_1 x + p_2)}$$

Decay too quickly

$$\lambda(t) = \int_0^t \left(\frac{t_0}{t_0 + x}\right)^{-(2p_1 x + p_2)} dx$$

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

# Thank You

# Q&A

饮 水 思 源　爱 国 荣 校