Supporting Information for

Decoupling Shape Retention from Polymerization Kinetics Enables Ambient-Temperature 3D Printing of Polystyrene

Vanessa Rosciardi[ǂ] [a], Federico Serpe[ǂ] [b,c], Luca Scozzafava [b], Yurii Promovych [c], Marco Costantini [c], Marco Sasso [d], Carlo Sabbatini [d], Roberta Angelini* [a], Andrea Barbetta* [b]

ǂ: Vanessa Rosciardi and Federico Serpe equally contributed to this article

a: Institute for Complex Systems – National Research Council (ISC-CNR) and Physics Department, “La Sapienza” University of Rome, Piazzale Aldo Moro 5, 00185 Rome, Italy

b: Department of Chemistry, “La Sapienza” University of Rome, Piazzale Aldo Moro 5, 00185 Rome, Italy

c: Institute of Physical Chemistry – Polish Academy of Sciences, Marcina Kasprzaka 44/52,01224 Warsaw, Poland

d: Department of Industrial Engineering and Mathematical Sciences, Polytechnic University of Marche, via Brecce Bianche 12, 60131 Ancona, Italy, Italy

Correspondence must be addressed to:

Andrea Barbetta: andrea.barbetta@uniroma1.it; Roberta Angelini: roberta.angelini@cnr.it

Table of Contents

Description of Supplementary Videos

Supplementary Video 1

Illustrative video of the 3D printing process.

Supplementary Video 2

Close-up video of the 3D printing process of an inward overhanging construct.

Supplementary Video 3

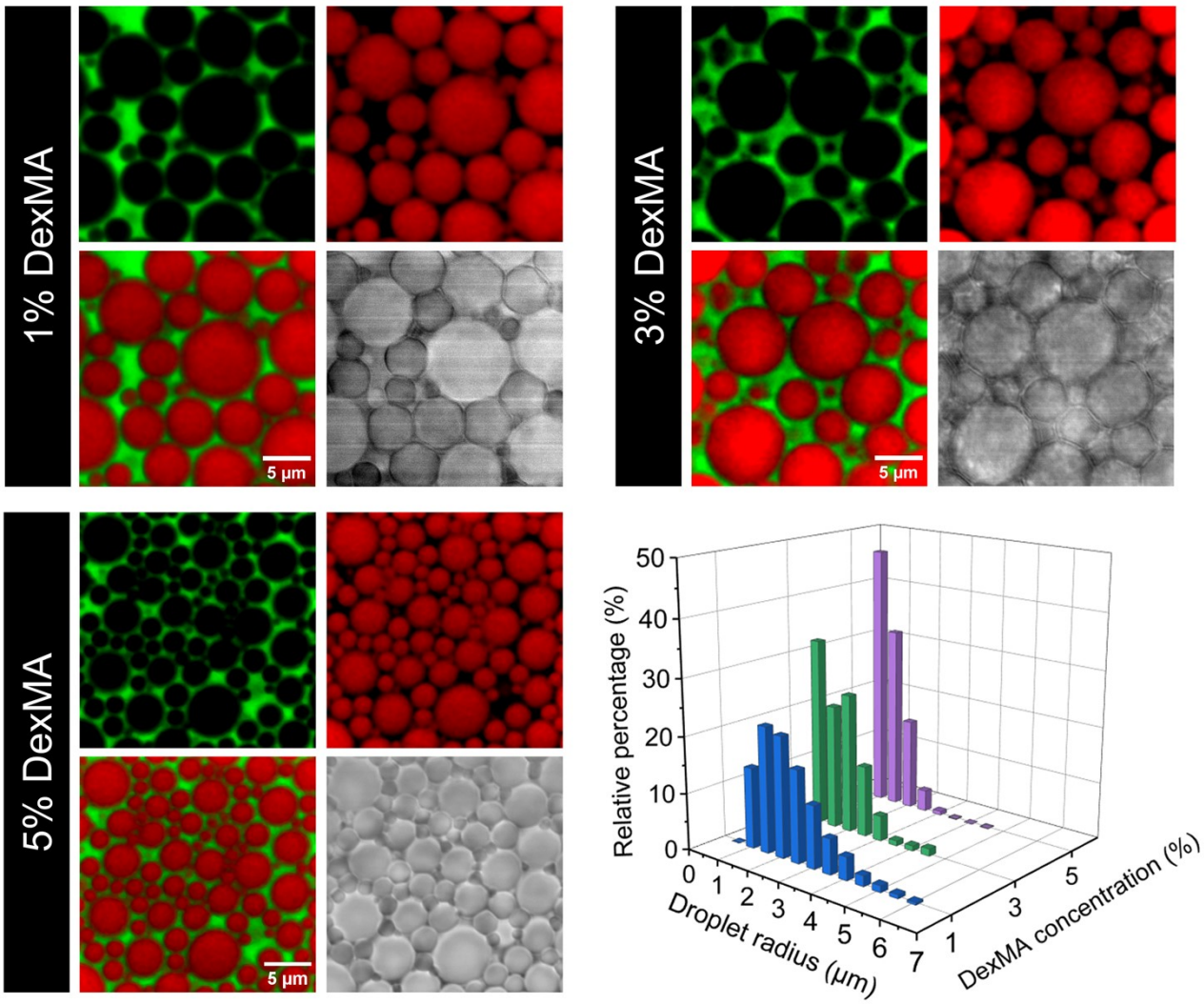
Close-up video of the 3D printing process of an outward overhanging construct.

Supplementary Video 4

Video of the complete 3D printing process of a hollow sphere.

Morphological and dimensional analysis

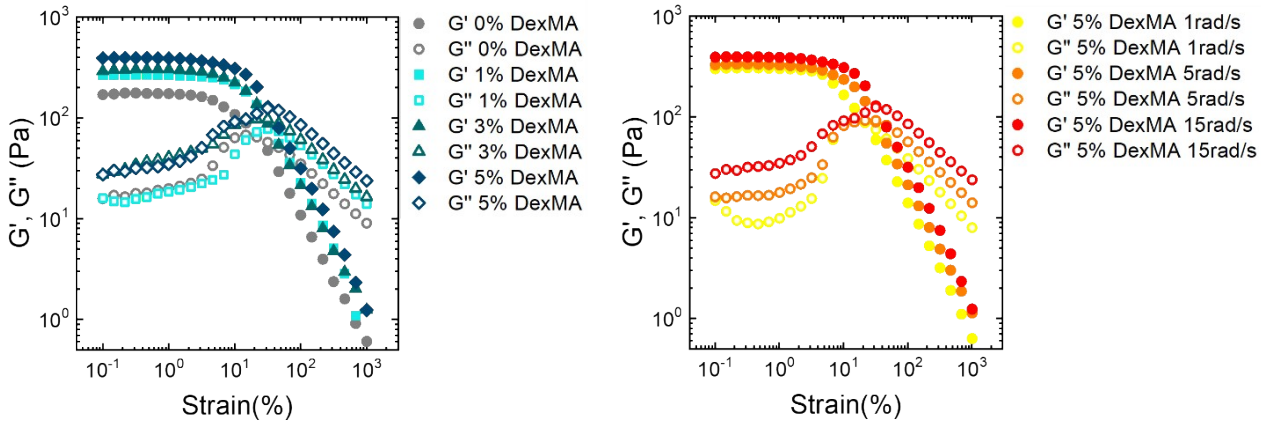
Morphological analysis of the HIPEs was conducted on fluorophore-containing samples via confocal laser scanning microscopy (CLSM). 2D images were analyzed to assess the dimensional distribution of the oil phase and its dependency on the methacrylated dextran (DexMA) concentration.



Supporting Figure 1. Morphological characterization of DexMA-based samples (1%, 3%, and 5% wt/v). Confocal laser scanning microscopy (CLSM) images show FITC–Dextran (green, aqueous continuous phase) and Nile Red (red, oil phase) in separate and merged channels. Grayscale panels correspond to transmitted light microscopy. **Top left**: 1% wt/v DexMA; **top right**: 3% wt/v DexMA; **bottom left**: 5% wt/v DexMA. **Bottom right**: Droplet size analysis from CLSM images, reporting average radii, size distributions, and their dependence on DexMA concentration.

Amplitude sweep tests

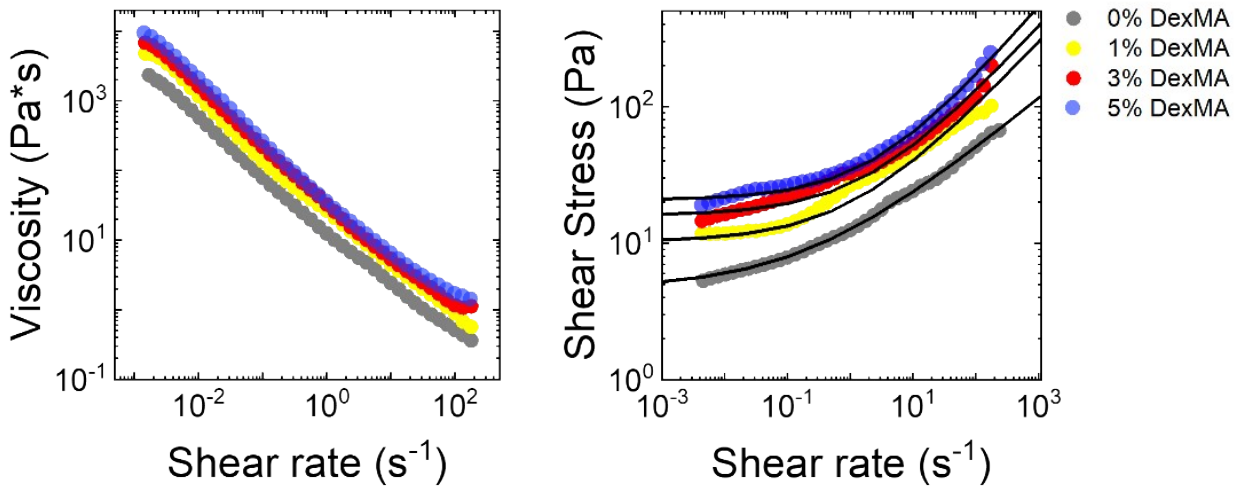
Rheological characterization was performed using an Anton Paar MCR302 rheometer equipped with a cone–plate geometry (cone angle: 2°, diameter: 25 mm). Oscillatory tests were performed to determine the storage modulus (*G′*) and loss modulus (*G″*) within the linear viscoelastic region (LVE) and to identify deviations from linearity and crossover points. These tests were carried out at angular frequencies of 1, 5, and 15 rad·s^{-1} with strain amplitudes varying from 0.1% to 1000%, under constant normal force. All measurements were performed in triplicate, and results are reported as mean values.



Supporting Figure 2. Rheological characterization in amplitude sweep mode. **Left:** Storage modulus (*G′*) and loss modulus (*G″*) as a function of strain for formulations with varying DexMA concentrations (0–5% wt/v). *G′* increases with DexMA content, and the *G′/G″* crossover shifts to higher strain values at higher concentrations. **Right:** Effect of frequency on *G′* and *G″* for 5% wt/v DexMA. Both moduli increase with frequency, while the crossover strain remains unchanged.

Steady-state viscosity tests

Rheological characterization was performed using an Anton Paar MCR302 rheometer equipped with a cone–plate geometry (cone angle: 2°, diameter: 25 mm). Shear-rate-controlled viscosity measurements were conducted over a range of 0.001–1000 s^{-1} following a pre-shear at 500 s^{-1}.



Supporting Figure 3. Rheological characterization in rotational mode. **Left:** Viscosity as a function of shear rate for samples with DexMA concentrations from 0% wt/v to 5% wt/v, showing pronounced shear-thinning behavior and increasing viscosity with higher DexMA content. **Right:** Shear stress versus shear rate for the same samples, highlighting yield stress values that increase with DexMA concentration. Experimental data (colored symbols) are fitted using the Herschel–Bulkley model (black lines).

LAOS Data Analysis

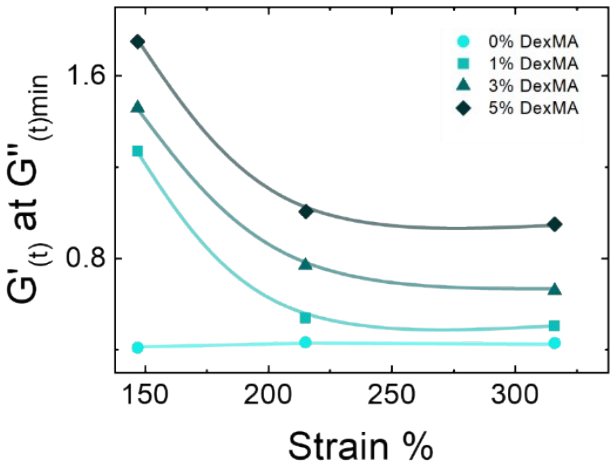
The non-linear rheological response was analyzed using a custom Python script. Input data consisted of CSV files with four columns: time, time-resolved shear strain, shear stress, and shear rate. The script generates 2D Lissajous plots (stress–strain and stress–rate) and their 3D projection in stress–strain–rate space. Experimental curves are interpolated and smoothed via fast Fourier transform (FFT). Frenet–Serret vectors are then constructed on the smoothed curve to calculate transient moduli, displacement stress, equilibrium strain, and derivatives of transient moduli. Output includes processed data (CSV) and plots: Lissajous curves, 3D deformation trajectories, Cole–Cole plots, and displacement stress versus time. The full script is provided below and is also available on GitHub.

```python
import os
import numpy as np
import pandas as pd
from scipy.fft import fft, ifft
from scipy.signal import savgol_filter
import plotly.graph_objects as go
import matplotlib.pyplot as plt

# ----------------------------
# 1. Create output folder on Desktop
# ----------------------------
desktop_path = os.path.join(os.path.expanduser("~"), "Desktop")
output_folder = os.path.join(desktop_path, "SPP_analysis")
os.makedirs(output_folder, exist_ok=True)

# ----------------------------
# 2. Load and preprocess data
# ----------------------------
file_path = "data prova 20_shear rate.CSV"  # Replace with your file path
df = pd.read_csv(file_path, skiprows=2, sep=';')
df.columns = ['Time', 'Strain', 'Stress', 'Rate']
df['Time'] -= df['Time'].min()

# Extract signals
time = df['Time'].values
strain = df['Strain'].values
rate = df['Rate'].values
stress = df['Stress'].values

# Interpolate to uniform time steps
uniform_time = np.linspace(time.min(), time.max(), len(time))
strain_interp = np.interp(uniform_time, time, strain)
rate_interp = np.interp(uniform_time, time, rate)
stress_interp = np.interp(uniform_time, time, stress)

# ----------------------------
# 3. FFT-based smoothing
# ----------------------------
def fft_smooth(signal, n_harmonics=15):
    N = len(signal)
    signal_fft = fft(signal)
    filtered_fft = np.zeros_like(signal_fft)
    filtered_fft[:n_harmonics] = signal_fft[:n_harmonics]
    filtered_fft[-n_harmonics+1:] = signal_fft[-n_harmonics+1:]
    return np.real(ifft(filtered_fft))

strain_smooth = fft_smooth(strain_interp)
rate_smooth = fft_smooth(rate_interp)
stress_smooth = fft_smooth(stress_interp)

# ----------------------------
# 4. Frenet-Serret frame & SPP metrics
```

```python
# ------------------------------
omega = 2 * np.pi / (uniform_time[-1] - uniform_time[0])
trajectory = np.vstack([strain_smooth, rate_smooth / omega, stress_smooth]).T

dt = uniform_time[1] - uniform_time[0]
dA = np.gradient(trajectory, axis=0)
T = dA / np.linalg.norm(dA, axis=1)[:, None]
dT = np.gradient(T, axis=0)
N = dT / np.linalg.norm(dT, axis=1)[:, None]
B = np.cross(T, N)

Gp_t = -B[:, 0] / B[:, 2]
Gpp_t = -B[:, 1] / B[:, 2]

sigma_d = stress_smooth - Gp_t * strain_smooth + Gpp_t * rate_smooth / omega
gamma_eq = stress_smooth / Gp_t
gamma_rec = strain_smooth - gamma_eq

Gp_dot = savgol_filter(np.gradient(Gp_t, dt), 15, 3)
Gpp_dot = savgol_filter(np.gradient(Gpp_t, dt), 15, 3)

# ------------------------------
# 5. Visualization and Save
# ------------------------------
def save_plot(fig, filename):
    fig_path = os.path.join(output_folder, filename)
    fig.write_image(fig_path)
    print(f"Saved plot: {fig_path}")

# Cole-Cole plot
fig1 = go.Figure()
fig1.add_trace(go.Scatter(x=Gp_t, y=Gpp_t, mode='lines', name='Cole-Cole'))
fig1.update_layout(title='Cole-Cole Plot', xaxis_title="G'_t", yaxis_title="G''_t")
save_plot(fig1, "cole_cole_plot.png")

# Transient moduli vs time
fig2 = go.Figure()
fig2.add_trace(go.Scatter(x=uniform_time, y=Gp_t, mode='lines', name="G'_t"))
fig2.add_trace(go.Scatter(x=uniform_time, y=Gpp_t, mode='lines', name="G''_t"))
fig2.update_layout(title='Transient Moduli vs Time', xaxis_title='Time (s)',
yaxis_title='Moduli')
save_plot(fig2, "transient_moduli.png")

# Derivatives of moduli
fig3 = go.Figure()
fig3.add_trace(go.Scatter(x=uniform_time, y=Gp_dot, mode='lines', name="dG'_t/dt"))
fig3.add_trace(go.Scatter(x=uniform_time, y=Gpp_dot, mode='lines', name="dG''_t/dt"))
fig3.update_layout(title='Derivatives of Transient Moduli', xaxis_title='Time (s)',
yaxis_title='Derivative')
save_plot(fig3, "moduli_derivatives.png")

# Displacement stress
fig4 = go.Figure()
fig4.add_trace(go.Scatter(x=uniform_time, y=sigma_d, mode='lines', name="Displacement
Stress"))
fig4.update_layout(title='Displacement Stress vs Time', xaxis_title='Time (s)',
yaxis_title='Stress (Pa)')
save_plot(fig4, "displacement_stress.png")

# Elastic Lissajous
fig5 = go.Figure()
fig5.add_trace(go.Scatter(x=strain_smooth, y=stress_smooth, mode='lines', name="Elastic
Lissajous"))
fig5.update_layout(title='Elastic Lissajous Plot', xaxis_title='Strain',
yaxis_title='Stress (Pa)')
save_plot(fig5, "elastic_lissajous.png")

# Viscous Lissajous
fig6 = go.Figure()
```

```
fig6.add_trace(go.Scatter(x=rate_smooth, y=stress_smooth, mode='lines', name="Viscous Lissajous"))
fig6.update_layout(title='Viscous Lissajous Plot', xaxis_title='Rate (1/s)', yaxis_title='Stress (Pa)')
save_plot(fig6, "viscous_lissajous.png")

# Waveform comparison
fig7 = go.Figure()
fig7.add_trace(go.Scatter(x=uniform_time, y=stress_interp, mode='lines', name="Original Stress"))
fig7.add_trace(go.Scatter(x=uniform_time, y=stress_smooth, mode='lines', name="Smoothed Stress"))
fig7.update_layout(title='Waveform Comparison', xaxis_title='Time (s)', yaxis_title='Stress (Pa)')
save_plot(fig7, "waveform_comparison.png")

# ------------------------------
# 6. Export processed data
# ------------------------------
processed_df = pd.DataFrame({
    'Time': uniform_time,
    'Strain': strain_smooth,
    'Rate': rate_smooth,
    'Stress': stress_smooth,
    "G'_t": Gp_t,
    "G''_t": Gpp_t,
    "dG'_t/dt": Gp_dot,
    "dG''_t/dt": Gpp_dot,
    "Displacement Stress": sigma_d,
    "Equilibrium Strain": gamma_eq,
    "Recoverable Strain": gamma_rec
})
csv_path = os.path.join(output_folder, "spp_processed_data.csv")
processed_df.to_csv(csv_path, index=False)
print(f"Saved processed data: {csv_path}")

# Harmonics via FFT
stress_fft = fft(stress_interp)
harmonics_df = pd.DataFrame({
    'Harmonic': np.arange(len(stress_fft)),
    'Magnitude': np.abs(stress_fft),
    'Phase': np.angle(stress_fft)
})
harmonics_path = os.path.join(output_folder, "harmonics_data.csv")
harmonics_df.to_csv(harmonics_path, index=False)
print(f"Saved harmonics data: {harmonics_path}")

# ------------------------------
# 7. Display plots using matplotlib
# ------------------------------
def display_plot(x, y, title, xlabel, ylabel):
    plt.figure()
    plt.plot(x, y)
    plt.title(title)
    plt.xlabel(xlabel)
    plt.ylabel(ylabel)
    plt.grid(True)
    plt.show()

display_plot(Gp_t, Gpp_t, "Cole-Cole Plot", "G'_t", "G''_t")
display_plot(uniform_time, Gp_t, "G'_t vs Time", "Time (s)", "G'_t")
display_plot(uniform_time, Gpp_t, "G''_t vs Time", "Time (s)", "G''_t")
display_plot(uniform_time, sigma_d, "Displacement Stress vs Time", "Time (s)", "Stress (Pa)")
display_plot(strain_smooth, stress_smooth, "Elastic Lissajous", "Strain", "Stress (Pa)")
display_plot(rate_smooth, stress_smooth, "Viscous Lissajous", "Rate (1/s)", "Stress (Pa)")
```

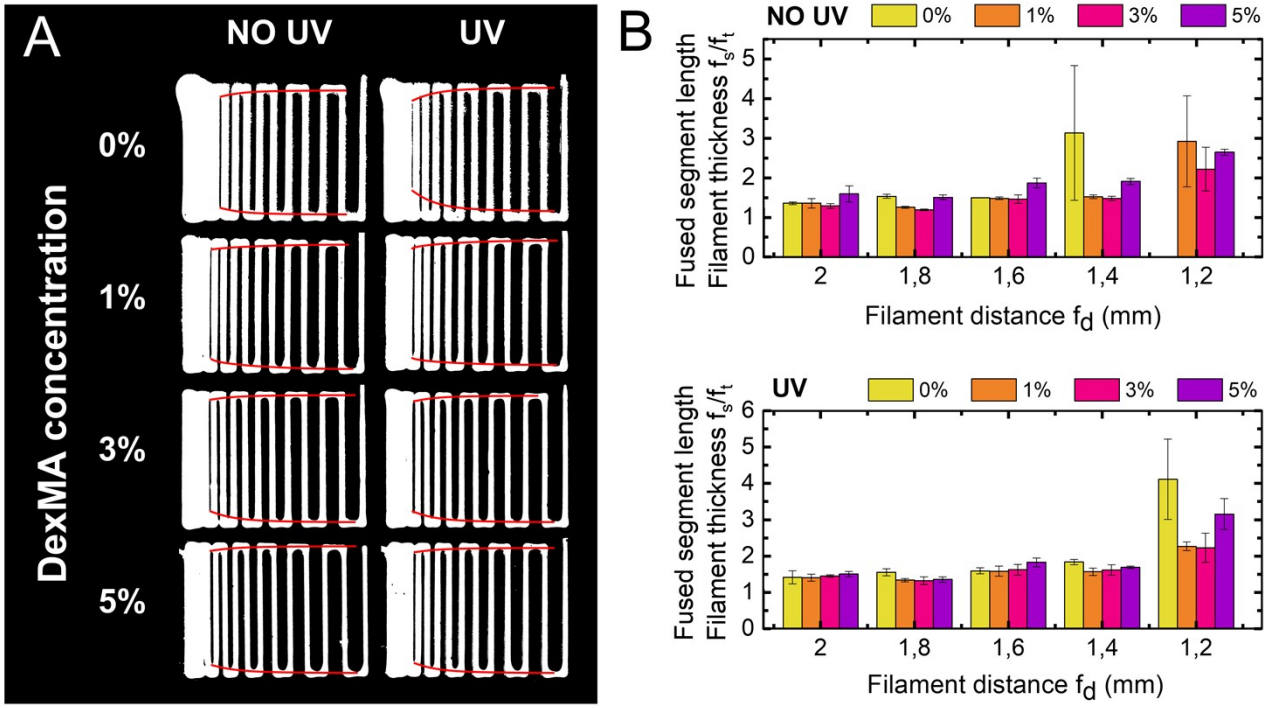
Residual elasticity (close-up)



Supporting Figure 4. Close-up of the dependence of $G'_{(t)}$ at $G''_{(t)min}$ at on applied strain for samples with varying DexMA concentrations (wt/v). Positive values indicate residual elasticity even at the maximum fluidized state. The curves approach a plateau at high strain, with plateau values increasing with DexMA content.

2D printing resolution

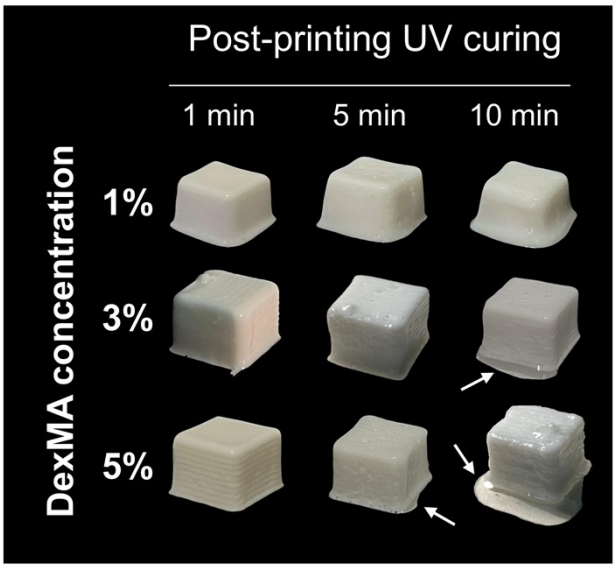
To evaluate the printability and structural fidelity of DexMA-based formulations, grids were printed under different conditions and analyzed for filament fusion. The effect of DexMA concentration and UV curing on inter-filament adhesion was assessed both qualitatively (optical images) and quantitatively (fusion ratio as a function of filament spacing). These tests provide insight into how formulation and post-processing influence dimensional accuracy in extrusion-based printing.



Supporting Figure 5. Printability analysis of DexMA-based formulations. (A) Optical images of printed grids at varying DexMA concentrations (0–5% wt/v) under two conditions: without UV curing (NO UV) and with UV curing (UV). (B) Quantification of filament fusion, expressed as fused segment length normalized by filament thickness (s_f/t_f), as a function of filament distance (f_d).

Oil leaking due to excessive UV curing

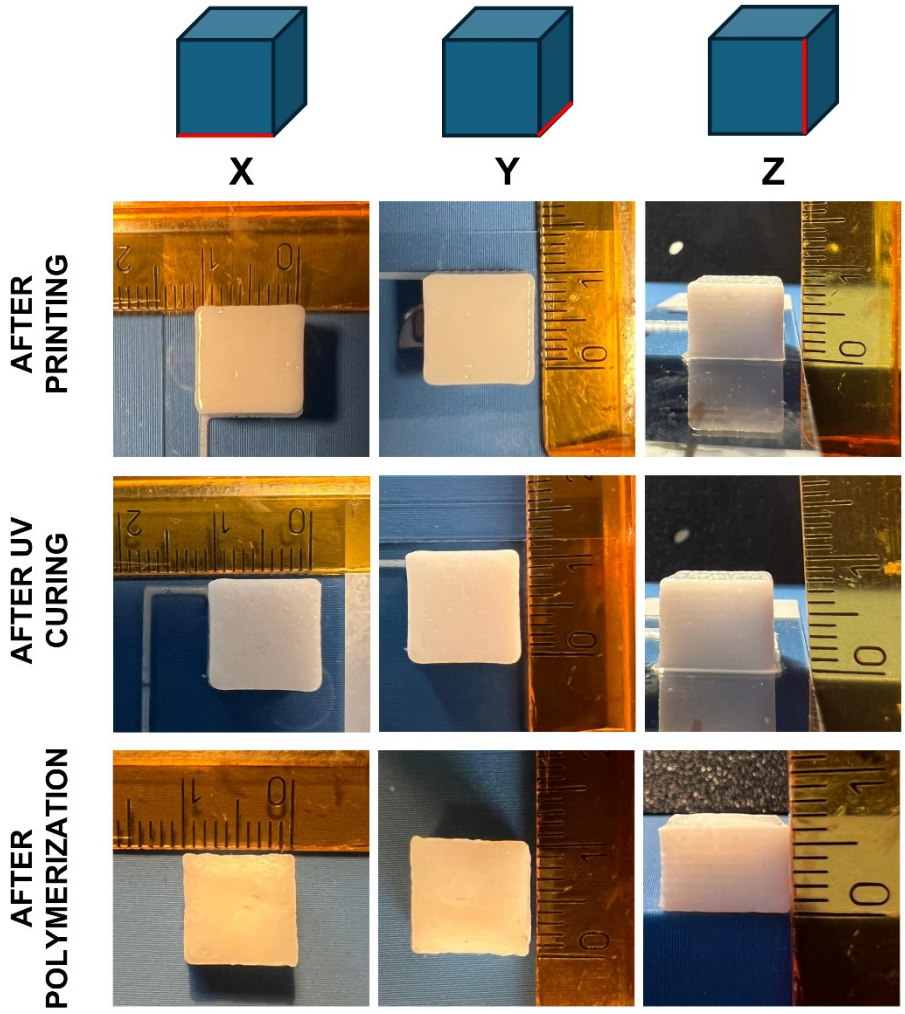
To evaluate the effect of post-printing UV curing on the structural integrity of the printed constructs, samples containing different DexMA concentrations (1%, 3%, and 5%) were exposed to UV light for varying durations. Prolonged UV exposure promotes excessive crosslinking, which leads to shrinkage and expulsion of the internal oil phase, compromising the material's stability.



Supporting Figure 6. Effect of post-printing UV curing time on printed constructs with varying DexMA concentrations (1%, 3%, and 5% wt/v). Samples were exposed to UV light for 1, 5, and 10 minutes. Extended curing (10 min) induces excessive crosslinking and oil phase leakage (indicated by arrows), particularly at higher DexMA concentrations.

Shrinkage of the 3D printed cubes

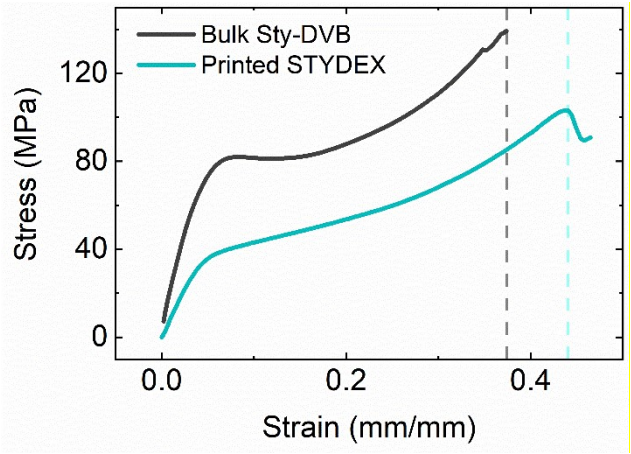
To estimate the shrinkage of 3D printed constructs, we conducted length measurements on each dimension of the simplest printed form (cubes) right after printing, after UV post-printing curing, and after 6 hours at 60 °C. As visible in the picture, dimensional loss along the X and Y axes cannot be appreciated with this measurement system and is therefore considered zero, while along the Z axis we observed a loss of 0.2 mm, likely due to effects in which gravity is the dominant component.



Supporting Figure 7. Photographs of the 3D printed constructs at different fabrication stages. A ruler was used as a refence to qualitatively assess dimensional loss for each dimension.

Mechanical compression tests

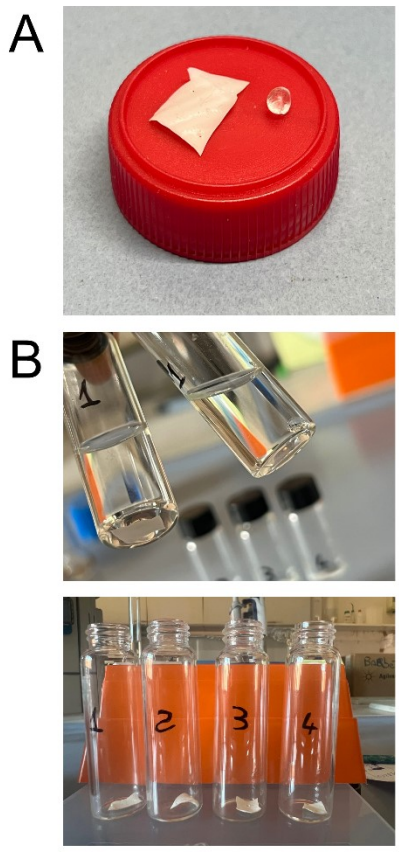
Compressive behavior was evaluated for printed STYDEX constructs and compared to bulk-polymerized poly(styrene-co-divinylbenzene) prepared under identical monomer and initiator ratios. Nominal stress–strain curves reveal distinct differences in deformation and failure modes.

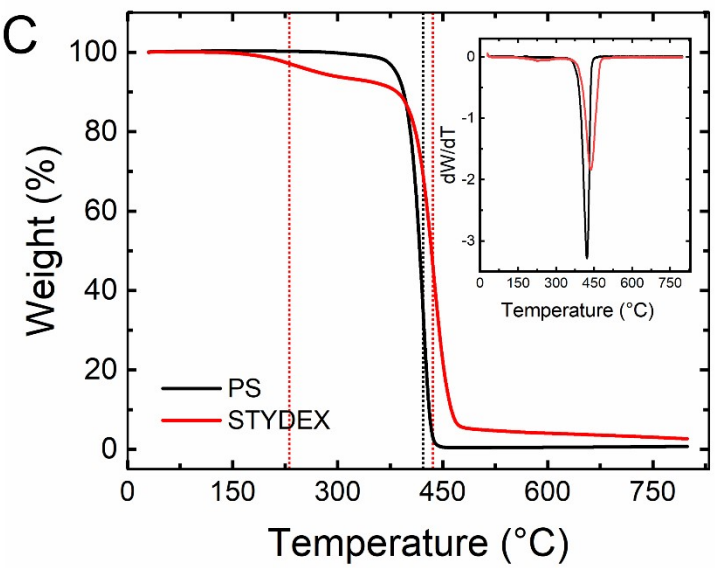


Supporting Figure 8. Compressive stress-strain curves for bulk poly(styrene-co-divinylbenzene) (Bulk Sty-DVB, gray) and a cubic specimen printed with the 5% DexMA HIPE formulation (Printed STYDEX, blue). Both materials exhibit strain hardening after the initial elastic region; however, the bulk sample reaches a higher maximum stress (>120 MPa) and shows a distinct yield plateau, while the printed construct displays a smoother transition and fails at ~90 MPa. Vertical dashed lines indicate strain at failure for each sample. Average elastic moduli (E), calculated from the initial linear region, were 1663 MPa for the bulk and 878 MPa for the printed sample. Despite the printed material having half the stiffness, its thoughness (area under the curve) was only ~25% lower, and its maximum strain at failure was ~15% higher than the bulk material. These results suggest that the hierarchical architecture of the printed samples promotes more efficient stress redistribution and energy dissipation.

Thermal stability and solvent resistance

The thermal stability and solvent resistance of the printed constructs were compared with those of commercial bulk linear polystyrene. Thermal stability was measured as described in the experimental section and highlighted a higher decomposition temperature for the printed cross-linked constructs (436 °C) compared to bulk linear polystyrene (422 °C). At lower temperatures (231 °C), the decomposition of DexMA is observed. Solvent resistance was evaluated using acetone, the best solvent for linear polystyrene. While the latter dissolved completely after 24 h, the cross-linked samples remained unaltered, as determined by the absence of measurable weight loss.





D

Initial weight	Final weight	Weight loss
0,02552 g	0,02528 g	0,94%
0,01569 g	0,01614 g	-2,86%
0,01962 g	0,02009 g	-2,39%
0,01746 g	0,01697 g	2,81%
Average weight loss		**-0,375%**

Supporting Figure 9. Thermal and solvent resistance evaluation. (A) Photograph of a fragment of the cross-linked sample and a commercial polystyrene pellet undergoing a thermogravimetric experiment. (B) Photographs of the dissolution of the two samples in acetone immediately after immersion (top) and after 24 hours (bottom). (C) Thermogravimetric curves of linear polystyrene (black, PS) and cross-linked samples (red, STYDEX). In the inset, the derivative of the thermogravimetric curves highlights the inflection point. (D) Table summarizing the weight loss, expressed in absolute and percentage values, of samples dissolved in acetone.