Electronic Supplementary Information

Optimising reaction conditions in flasks for performances in organic light-emitting devices

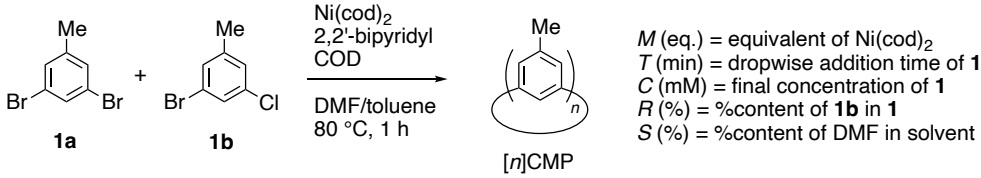
Koki Ikemoto, Misato Akiyoshi, Ayano Kobayashi, Hiroshi Kita, Hideo Taka and Hiroyuki Isobe

Table of contents

Materials

Dibromotoluene **1a**, bromochlorotoluene **1b** and 1,5-cyclooctadiene were purchased from TCI. 2,2'-bipyridyl was purchased from BLD pharmatech Ltd. Bis(1,5-cyclooctadiene)nickel, anhydrous DMF and anhydrous toluene were purchased from Kanto. Anhydrous DMF and toluene were further purified by a solvent purification system (GlassContour) equipped with columns of activated alumina and supported copper catalyst (Q-5).[1]

Reactions in Flasks



For the optimisation of macrocyclisation to [*n*]CMPs with **1**, we decided to examine five factors, equivalent of $Ni(cod)_2$ (*M*), dropwise addition time of **1** (*T*) and final concentration of **1** (*C*), %content of bromochlorotoluene (**1b**) in **1** (*R*) and %content of DMF in solvent (*S*), and each factor was examined with 3 levels (Fig. S1). To include these five factors with three levels, the L_{18} ($2^1 \times 3^7$) table in Taguchi's orthogonal array was referred for the DoE settings.[2] The five factors, (*M*, *T*, *C*, *R*, *S*), defined a five-dimensional parameter space, which can be depicted with three-dimensional (*M*, *T*, *C*) diagrams located in a two-dimensional (*R*, *S*) space (Fig. S1). As shown in Fig. S1, reactions to be performed are evenly distributed in the five-dimensional parameter space. We carried out these 18 reactions under the designed conditions, and to validate the model obtained by machine learning (see below), two additional reactions were carried out at (*M*, *T*, *C*, *R*, *S*) = (2, 9, 64, 5, 33) and (2, 11, 50, 8, 29). A typical experimental procedure is described for the condition of (*M*, *T*, *C*, *R*, *S*) = (2, 9, 64, 5, 33) as an example. A mixture of bis(1,5-cyclooctadiene)nickel (660 mg, 2.40 mmol), 2,2'-bipyridyl (375 mg, 2.40 mmol) and 1,5-cyclooctadiene (0.310 ml, 2.40 mmol) in DMF/toluene (3.1 mL/3.1 mL) was stirred at 80 °C for 30 min. To the mixture was added a solution of dibromotoluene **1a** (285 mg, 1.14 mmol) and bromochlorotoluene **1b** (12.3 mg, 0.0600 mmol) in DMF/toluene (3.2 mL/9.4 mL) dropwise over 9 min by using a syringe pump (YMC YSP-101). The mixture was further stirred at 80 °C for 1 h. After the reaction mixture was cooled down to ambient temperature, 1 M aq. HCl (30 mL) was added, and the mixture was vigorously stirred overnight. The aqueous layer was extracted with $CHCl_3$ (20 mL × 3), and the combined organic layer was washed with brine (50 mL), dried over Na_2SO_4 and concentrated in vacuo. The residue was dissolved in $CHCl_3$ (5 mL) and passed through a pad of silica gel (20 mL), and additional $CHCl_3$ (50 mL) was passed to obtain a crude raw mixture of methylated [*n*]CMPs as a white powder (154 mg). The crude raw materials of methylated [*n*]CMP congeners were directly used to fabricate the double-layer OLED. The samples were also analysed by MALDI-TOF MS (Bruker Daltonics autoflex speed; 0.1 mg specimen + 45 mg

tetracyanoquinodimethane) under a reflector positive mode with 30% laser power (Figs. S2 and S3) and ^{1}H NMR spectroscopy (JEOL JNM-ECA II 600 equipped with an UltraCOOL probe; 600 MHz, $CDCl_3$, 298 K) using bromoform as an internal standard (Fig. S4).

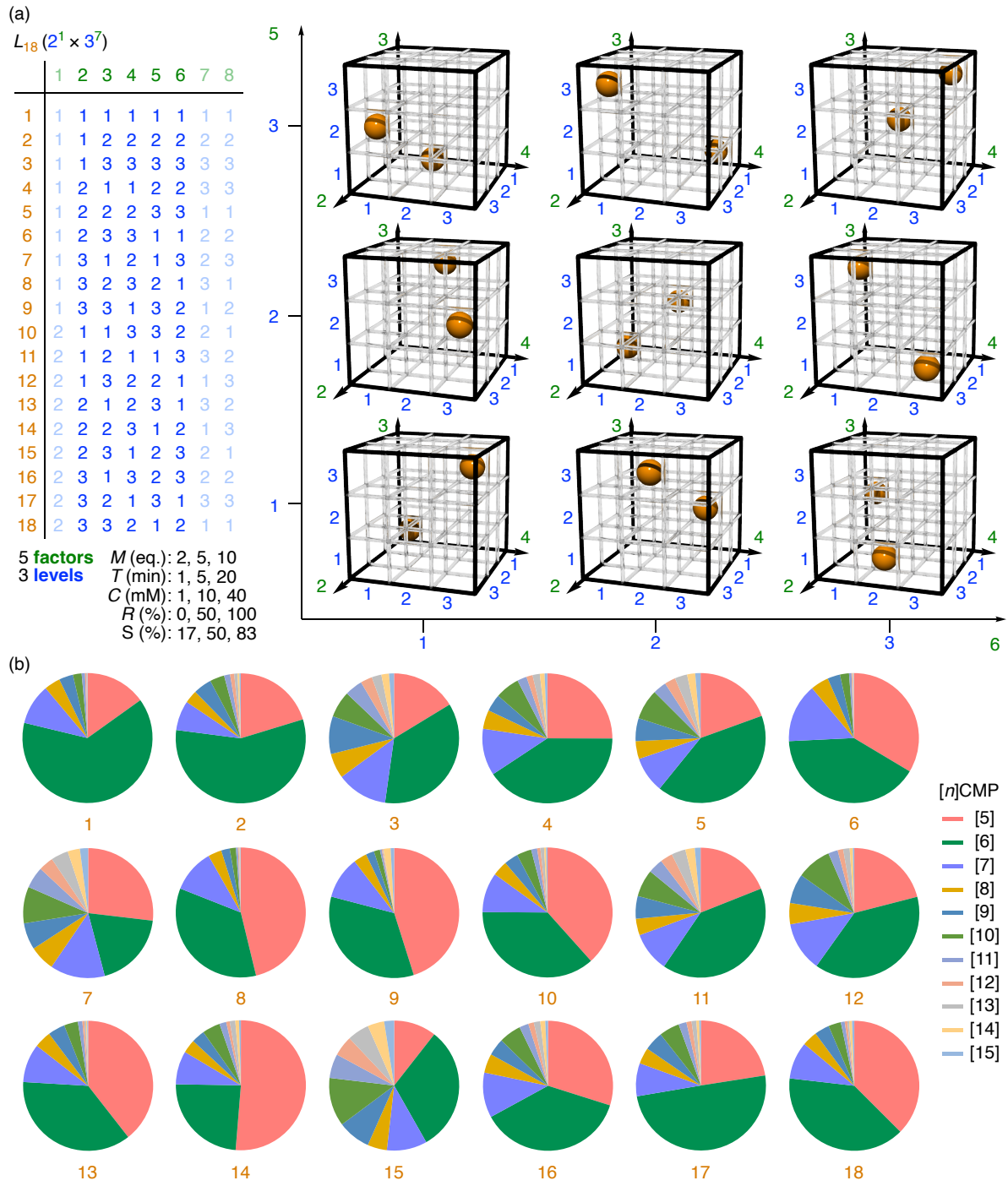


Fig. S1 Design of the mixture of methylated [*n*]CMP. (a) $L_{18}(2^1 \times 3^7)$ orthogonal table and locations of the designed condition in a five-dimensional parameter space of (*M*, *T*, *C*, *R*, *S*). Actual experimental data in the five-dimensional space are shown in Fig. S6a. (b) Population of methylated [*n*]CMP congeners in the 18 crude raw materials obtained by DoE-guided experiments. Populations were estimated by using the signal intensities of the MALDI spectra shown in Fig. S2.

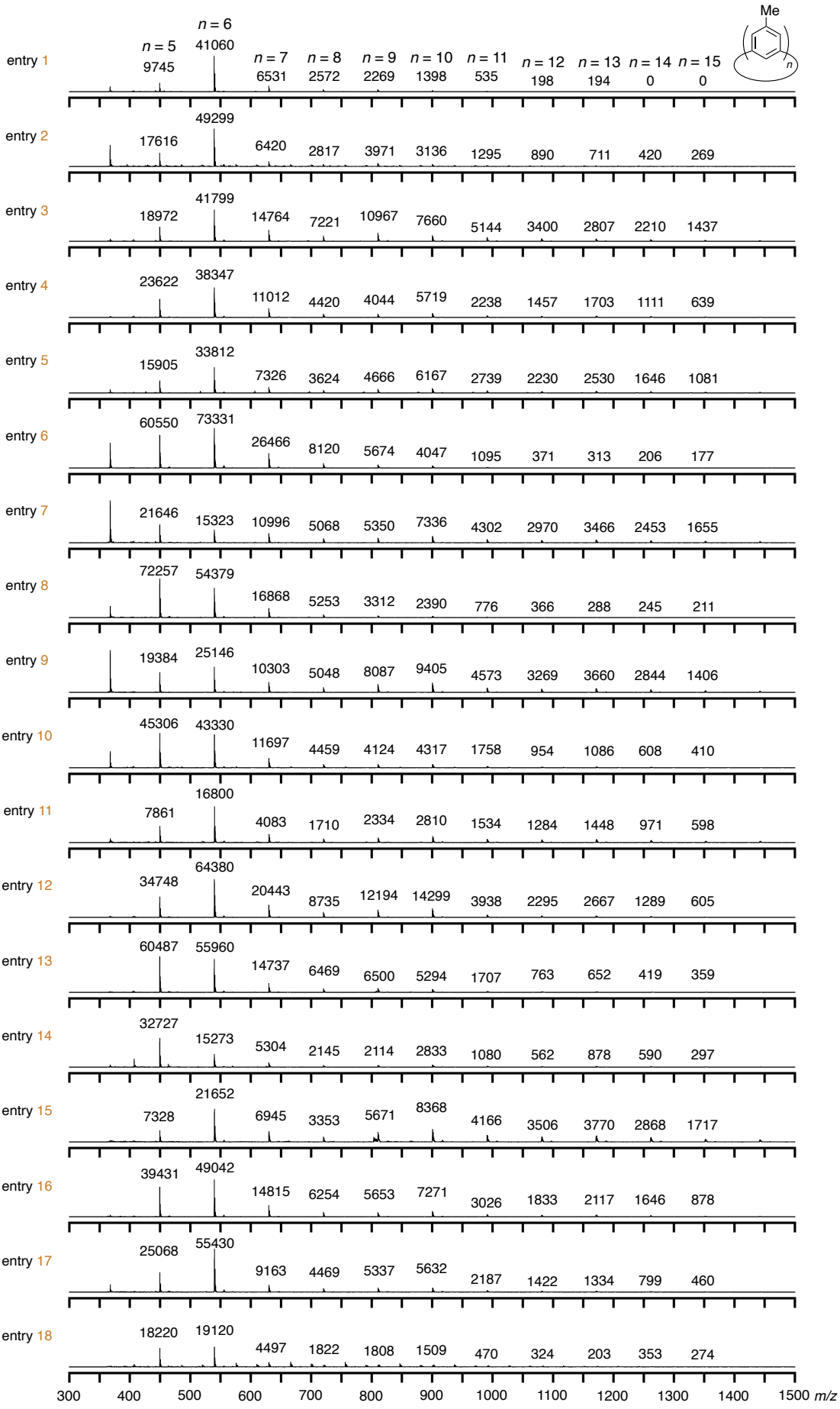


Fig. S2 MALDI-TOF MS spectra of the 18 crude raw materials obtained by DoE-guided experiments.

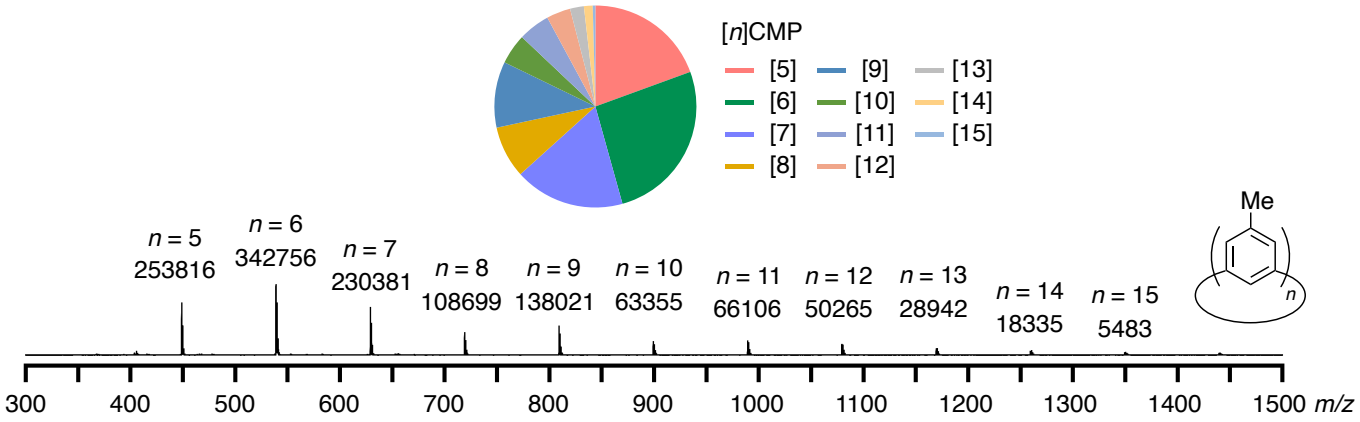


Fig. S3 MALDI-TOF MS spectrum of methylated [*n*]CMPs@(*M*, *T*, *C*, *R*, *S*) = (2, 9, 64, 5, 33). Based on the signal intensities shown in the spectrum, we estimated the populations of [*n*]CMP congeners.

	NMR yield (%)				
	[5]CMP	[6]CMP	[7]CMP	NMR	MS
1	4	16	4		
2	2	8	2		
3	4	15	5		
4	11	17	10		
5	2	6	3		
6	18	27	12		
7	9	12	10		
8	20	17	8		
9	2	10	4		
10	15	12	6		
11	2	5	3		
12	8	23	10		
13	23	25	8		
14	9	10	5		
15	1	8	5		
16	13	14	9		
17	9	21	9		
18	2	4	2		

Fig. S4 NMR yields of [5]-[7]CMP congeners and comparison of the NMR and MS populations of [5]-[7]CMP congeners in the 18 crude raw materials obtained by DoE-guided experiments. Because of the severe overlap of signals, the yields of [8]-[15]CMPs were not determined.[3]

Performances in Devices

Glass substrates with indium-tin oxide (ITO; 120 nm) was coated with a thin layer of poly(ethylenedioxy)thiophene:polystyrene sulfonate (PEDOT:PSS; 60 nm) by spin-coating of the aqueous dispersion. In a glove box, the glass substrate was coated with emission layer (EML) by spin-coating a solution of a crude raw mixture of methylated [*n*]CMPs (0.34 wt%) and **3** (0.06 wt%) in chlorobenzene and was baked at 120 °C for 30 min. The thickness of EML was measured as 20 nm by Dektak-XT (Bruker). Deposition of 1,3,5-tris(*N*-phenylbenzimidazol-2-yl)benzene (TPBi; **2**) via sublimation as electron transport layer (ETL; 60 nm) was followed by vapor deposition of LiF (2 nm) and Al (100 nm) on a fabrication instrument (SAL3000, ALS Technology) to furnish four devices (2 × 2 mm) on a single substrate (See Graphical Abstract of ref. 4). The current density-voltage characteristics and electroluminescence were measured on IZU-IS001S (System Engineers) and CS-2000 (Konica Minolta). External quantum efficiency (EQE), current efficiency (CE) and driving voltage (DV) were recorded at constant current of 2.5 mA•cm^{-2}. The performances to derive the ML models are shown in Table S1, and those obtained to validate the ML model are shown in Table S2. As reference OLEDs, we also fabricated the devices with pure [*n*]CMP (*n* = 5 and 6, respectively), and the performances are also shown in Table S2.

Table S1. Device performances of OLEDs at 2.5 mA•cm^{-2} for the ML models

[*n*]CMP	EQE (%)				CE (cd•A^{-1})				DV (V)			
(*M*, *T*, *C*, *R*, *S*)	1st	2nd	3rd	4th	1st	2nd	3rd	4th	1st	2nd	3rd	4th
(2, 1, 1, 0, 17)	1.4	1.3	1.4	1.4	4.8	4.5	4.7	4.8	5.2	5.1	5.1	5.2
(2, 5, 10, 50, 50)	2.4	2.4	2.4	2.4	8.0	8.0	8.0	8.0	5.8	5.9	5.8	5.7
(2, 20, 40, 100, 83)	5.3	5.3	5.5	5.4	18.2	18.1	18.8	18.5	6.1	6.1	6.1	6.2
(5, 1, 1, 50, 50)	2.4	2.3	2.3	2.3	7.9	7.9	7.9	7.9	6.3	6.4	6.5	6.4
(5, 5, 10, 100, 83)	3.0	2.9	2.9	3.0	10.1	9.9	10.0	10.3	6.4	6.2	6.4	6.5
(5, 20, 40, 0, 17)	9.0	9.0	8.8	8.9	31.0	31.0	30.4	30.5	5.9	5.8	5.8	5.8
(10, 5, 1, 100, 17)	2.9	1.5	2.6	2.8	9.5	4.7	8.9	9.4	5.4	4.7	5.4	5.4
(10, 20, 10, 0, 50)	2.7	2.7	2.7	2.7	9.2	9.0	9.1	9.1	6.8	6.7	6.7	6.7
(10, 1, 40, 50, 83)	5.0	4.9	4.9	4.9	17.3	17.0	17.2	17.0	6.5	6.5	6.5	6.5
(2, 20, 1, 50, 83)	1.6	1.5	1.6	1.6	5.1	5.0	5.1	5.2	6.7	6.7	6.7	6.6
(2, 1, 10, 100, 17)	5.5	5.4	5.4	5.4	18.8	18.4	18.4	18.5	6.1	6.1	6.2	6.1
(2, 5, 40, 0, 50)	8.5	8.3	8.4	8.6	29.5	28.6	28.9	29.4	5.6	5.6	5.6	5.6
(5, 5, 1, 0, 83)	0.4	0.5	0.8	0.8	1.5	1.7	2.5	2.5	6.5	6.5	6.3	6.4
(5, 20, 10, 50, 17)	4.9	4.9	4.9	4.9	16.7	16.4	16.5	16.5	6.3	6.2	6.2	6.3
(5, 1, 40, 100, 50)	8.2	7.9	8.0	8.0	28.0	26.8	27.5	27.7	6.1	6.2	6.1	6.1
(10, 20, 1, 100, 50)	4.4	4.4	4.3	4.4	14.9	15.1	15.0	14.6	6.1	6.1	6.0	6.1
(10, 1, 10, 0, 83)	2.1	2.0	2.0	2.0	6.9	6.7	6.7	6.7	6.9	6.7	6.8	6.8
(10, 5, 40, 50, 17)	5.3	5.3	5.2	5.2	17.5	17.3	17.3	17.6	5.3	5.1	5.2	5.3

Table S2. Device performances of OLEDs at 2.5 mA•cm^{-2} for model validations. Reference data with pure [*n*]CMP are also shown.

[*n*]CMP	EQE (%)				CE (cd•A^{-1})				DV (V)			
(*M*, *T*, *C*, *R*, *S*)	1st	2nd	3rd	4th	1st	2nd	3rd	4th	1st	2nd	3rd	4th
(2, 9, 64, 5, 33)	9.5	9.6	9.6	9.5	33.2	33.4	33.4	33.3	5.9	5.9	5.9	5.9
(2, 11, 50, 8, 29)	9.3	9.3	9.3	9.3	30.7	30.8	31.0	31.0	5.1	5.1	5.1	5.1
n = 5	0.9	0.8	0.8	0.9	3.0	2.6	2.5	2.8	5.1	5.0	5.0	5.1
n = 6	1.0	1.0	0.8	0.3	2.5	2.5	1.9	0.9	5.7	5.7	5.6	5.3

Machine-Learning Models

Based on 18 × 4 datasets of condition-EQE relationships shown in Table S1, the ML models to correlate the reaction with the EQE values were constructed, and corresponding heatmap models were generated. Basic methods were similar to those reported in our previous studies,[5] and following modules on Python were used: scikit-learn (v. 0.24.2),[6] pandas data handling library (v. 1.2.4),[7] and matplotlib plotting library (v. 3.1.0).[8] Three ML methods were examined in this study: support vector regression (SVR), partial least squares regression (PLSR) and multilayer perceptron (MLP). Hyperparameter tunings for the ML models were performed by using the grid search protocol to minimize the mean squared errors (MSEs) during the leave-one-out cross validation (LOOCV) (Code S1-S3).[9] Initial settings for the hyperparameter tunings of each model as well as the optimised settings for the minimum MSE are shown below. The SVR model showed the smallest MSE and was found as the most appropriate ML model. In the five-dimensional SVR model, the maximum EQE value was searched (Code S4). A search range for each factor was set as $1 \leq M < 11$, $1 \leq T < 31$, $1 \leq C < 101$, $1 \leq R < 101$ and $1 \leq S < 101$, which were divided in 11, 31, 101, 101 and 101 grids for the search, respectively. The search found the highest EQE value of 11.3% at (*M*, *T*, *C*, *R*, *S*) = (2, 9, 64, 5, 33).

Support vector regression (SVR)

Initial settings (grid range): $1\times1 \leq C \leq 1\times21$, $0.01\times1 \leq \varepsilon \leq 0.01\times10$, $0.01\times1 \leq \gamma \leq 0.01\times10$

Optimal settings: $C = 9$, $\varepsilon = 0.02$, $\gamma = 0.09$ (MSE = 0.0368)

Partial least squares regression (PLSR)

Initial settings (grid range): $1\times1 \leq$ degree $\leq 1\times5$, $1\times1 \leq$ n components $\leq 1\times10$

Optimal settings: degree = 4, n components = 10 (MSE = 0.0396)

Multilayer perceptron (MLP)

Initial settings (grid range): hidden layer size: 10, 50, 100, 500, 1000

Optimal settings: hidden layer size = 100 (MSE = 0.2606)

The credibility of the ML models was also evaluated by a correlation diagram between experimental EQE values and ML-derived EQE values with R^2 (Fig. S5).

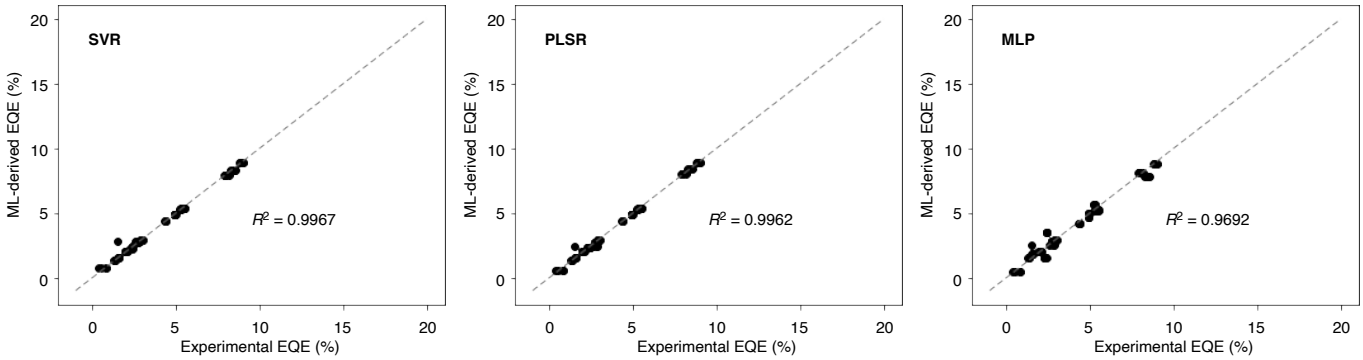


Fig. S5 Correlation diagrams between experimental EQE values and ML-derived EQE values. The credibility of the correlation was also evaluated by the determination coefficient, R^2.

The condition-EQE relationships were depicted in heatmap representations (Fig. S6). To gain overviews of the relationships, three-dimensional (*M*, *T*, *C*) diagrams were located in the two-dimensional space of (*R*, *S*). For the representative locations, actual experimental settings at (*R*, *S*) = (0, 17), (50, 17), (100, 17), (0, 50), (50, 50), (100, 50), (0, 83), (50, 83) and (100, 83) were adopted, and 3×3 (*M*, *T*, *C*) diagrams were created. As an example, a code for (*R*, *S*) = (50, 50) for the SVR model is shown below (Code S5): the ML model was first constructed under the optimal hyperparameter settings, and, in a (*M*, *T*, *C*) space, 11 layers of heatmaps composed of 50×50 points were then stacked to afford one (*M*, *T*, *C*) diagram. The 9 diagrams in total were created by changing (*R*, *S*) settings in the code.

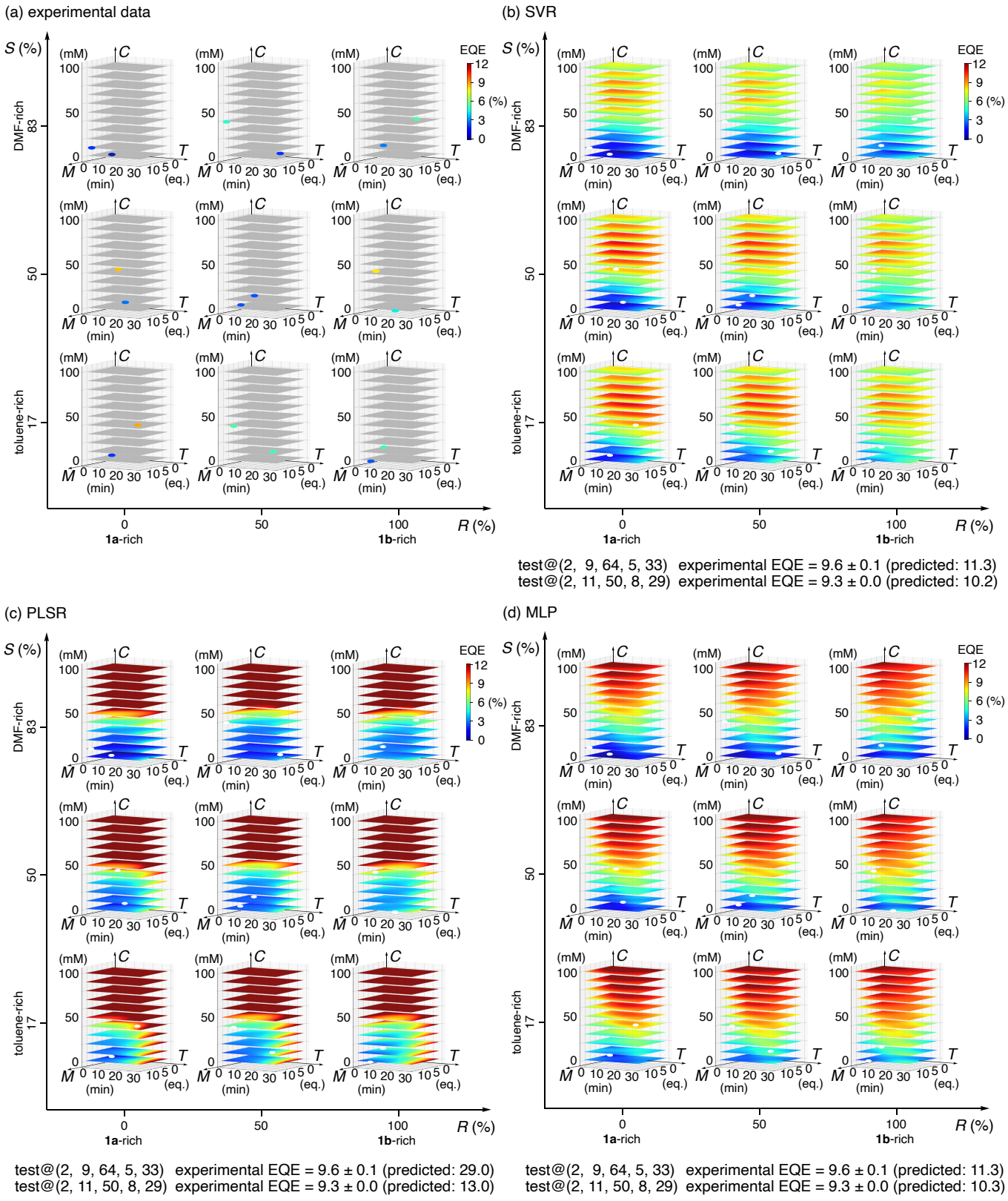


Fig. S6 Condition-EQE relationships depicted in the heatmaps in the five-dimension parameter space. (a) Experimental data. (b) SVR model. (c) PLSR model. (d) MLP model. The heatmaps at (R, S) = (50, 50) are also shown in Fig. 4.

In the following codes, the experimental datasets were loaded from an csv file, "mctsr_eqe.csv".

Code S1. Grid search with SVR

```
##SVR gridsearch
import pandas as pd
from sklearn import svm
from sklearn.model_selection import GridSearchCV, LeaveOneOut
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
from sklearn.metrics import r2_score

# Loading experimental data
df = pd.read_csv("mctsr_eqe.csv").dropna()
names = ('m', 'c', 't', 's', 'r')
X = df.loc[:, names].to_numpy()
y = df['eqe'].to_numpy()

# Pre-normalize the input X
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Ranges of hyperparameters for optimization
svm_params = {
    "C": [1*i for i in range(1, 21)],
    "gamma": [0.01*i for i in range(1, 11)],
    "epsilon": [0.01*i for i in range(1, 11)]
}

# Define and perform grid search with LOOCV.
grid_search = GridSearchCV(svm.SVR(), svm_params,
                           cv=LeaveOneOut(),
                           scoring="neg_mean_squared_error",
                           n_jobs=-1)
grid_search.fit(X_scaled, y)
print(f"Best model params: {grid_search.best_params_}")
print(f"Best model score: {grid_search.best_score_}")

# Construct a SVR model with the optimized parameters
bestestimator = svm.SVR(**grid_search.best_params_)
bestestimator.fit(X_scaled, y)
pred = bestestimator.predict(X_scaled)

# Plot predicted values against experimental ones
fig, ax1 = plt.subplots(1, 1, clear=True, figsize=(5, 4))
plt.scatter(y, pred, color='black')
plt.plot([-1, 20], [-1, 20], ls="--", c=".3")
ax1.set_ylabel('ML-derived EQE (%)')
ax1.set_xlabel('Experimental EQE (%)')
plt.tight_layout()
plt.show()
```

```
print(f"r2 {r2_score(y, pred)}")
```

Code S2. Grid search with PLSR

```
##PLSR Gridsearch
from sklearn.model_selection import GridSearchCV, LeaveOneOut
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler, PolynomialFeatures
from sklearn.cross_decomposition import PLSRegression
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.metrics import r2_score

# Loading experimental data
df = pd.read_csv("mctsr_eqe.csv").dropna()
names = ('m', 'c', 't', 's', 'r')
X = df.loc[:, names].to_numpy()
y = df['eqe'].to_numpy()

# Pre-normalize the input X
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Ranges of hyperparameters for optimization
plsr_params = {
    "poly__degree": [i for i in range(1, 6)],
    "plsr__n_components": [i for i in range(1,11)],
}

# Define and perform grid search with LOOCV.
grid_search = GridSearchCV(
    Pipeline([
        ("poly", PolynomialFeatures()),
        ("plsr", PLSRegression())]),
    plsr_params,
    cv=LeaveOneOut(),
    scoring="neg_mean_squared_error",
    n_jobs=-1)
grid_search.fit(X_scaled, y)
print(f"Best model params: {grid_search.best_params_}")
print(f"Best model score: {grid_search.best_score_}")

# Construct a PLSR model with the optimized parameters
bestestimator = Pipeline([("poly",
PolynomialFeatures(degree=grid_search.best_params_["poly__degree"])),
                          ("plsr",
PLSRegression(n_components=grid_search.best_params_["plsr__n_components"]))])
bestestimator.fit(X_scaled, y)
pred = bestestimator.predict(X_scaled)

# Plot predicted values against experimental ones
fig, ax1 = plt.subplots(1, 1, clear=True, figsize=(5, 4))
```

```
plt.scatter(y, pred, color='black')
plt.plot([-1, 20], [-1, 20], ls="--", c=".3")
ax1.set_ylabel('ML-derived EQE (%)')
ax1.set_xlabel('Experimental EQE (%)')
plt.tight_layout()
plt.show()
print(f"r2 {r2_score(y, pred)}")
```

Code S3. Grid search with MLP

```
##MLP Gridsearch
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import GridSearchCV, LeaveOneOut
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
from sklearn.metrics import r2_score
from sklearn.neural_network import MLPRegressor

# Loading experimental data
df = pd.read_csv("mctsr_eqe.csv").dropna()
names = ('m', 'c', 't', 's', 'r')
X = df.loc[:, names].to_numpy()
y = df['eqe'].to_numpy()

# Pre-normalize the input X
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Ranges of hyperparameters for optimization
nn_params = {
    "hidden_layer_sizes": [(i,) for i in [10, 50, 100, 500, 1000]]
}

# Define and perform grid search with LOOCV.
grid_search = GridSearchCV(MLPRegressor(activation="logistic", max_iter=30000),
    nn_params,
    cv=LeaveOneOut(),
    scoring="neg_mean_squared_error",
    n_jobs=-1)
grid_search.fit(X_scaled, y)
print(grid_search.best_params_)
print(grid_search.best_score_)

# Construct a MLP model with the optimized parameters
bestestimator = MLPRegressor(activation="logistic", max_iter=30000,
**grid_search.best_params_)
bestestimator.fit(X_scaled, y)
pred = bestestimator.predict(X_scaled)

# Plot predicted values against experimental ones
fig, ax1 = plt.subplots(1, 1, clear=True, figsize=(5, 4))
```

```
plt.scatter(y, pred, color='black')
plt.plot([-1, 20], [-1, 20], ls="--", c=".3")
ax1.set_ylabel('ML-derived EQE (%)')
ax1.set_xlabel('Experimental EQE (%)')
plt.tight_layout()
plt.show()
print(r2_score(y, pred))
```

Code S4: Search for the highest EQE value in the SVR model

```
# Importing libraries
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import matplotlib as mpl
import pandas as pd
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn import svm
from mpl_toolkits.mplot3d import Axes3D

# Loading experimental data
df = pd.read_csv("mctsr_eqe.csv")
df = df.dropna()
names = ('m', 'c', 't', 's', 'r')
variables = df.loc[:, names]

# Constructing a SVR model with the optimized hyperparameters
reg_cmp8 = Pipeline([('scl', StandardScaler()), ('clf', svm.SVR(kernel='rbf', C=9,
epsilon=0.02, gamma=0.09))])
reg_cmp8.fit(variables, df.eqe)
df['pred_svm_cmp8'] = reg_cmp8.predict(variables)
print(df['pred_svm_cmp8'])

# Search for max
x_len, y_len, z_len, s_len, r_len = 11, 101, 31, 101, 101
xs = np.linspace(1, 11, x_len) # M
ys = np.linspace(1, 101, y_len) # C
zs = np.linspace(1, 31, z_len)  # T
ss = np.linspace(1, 101, s_len)
rs = np.linspace(1, 101, r_len)

xm,ym,zm,sm,rm = np.meshgrid(xs, ys, zs, ss, rs)
r = np.c_[xm.flatten(), ym.flatten(), zm.flatten(), sm.flatten(), rm.flatten() ]
c = reg_cmp8.predict(r)

maxp = 0
for coord,pred in zip(r,c):
    if pred > maxp:
        maxp = pred
        maxc = coord
print(maxp, maxc)
```

Code S5. An example of heatmap creation. SVR@(R, S) = (50, 50)

```
# Importing libraries
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import matplotlib as mpl
import pandas as pd
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn import svm
from mpl_toolkits.mplot3d import Axes3D

# Loading experimental data
df = pd.read_csv("mctsr_eqe.csv")
df = df.dropna()
names = ('m', 'c', 't', 's', 'r')
variables = df.loc[:, names]

# Constructing a SVR model with the optimized hyperparameters
reg_cmp8 = Pipeline([('scl', StandardScaler()), ('clf', svm.SVR(kernel='rbf', C=9,
epsilon=0.02, gamma=0.09))])
reg_cmp8.fit(variables, df.eqe)
df['pred_svm_cmp8'] = reg_cmp8.predict(variables)
print(df['pred_svm_cmp8'])

# Drawing heatmaps
fig = plt.figure(figsize = (10,10))
ax = fig.add_subplot(111, projection = "3d", proj_type="ortho")
ax.set_box_aspect((1,1,1.5))

x_len, y_len = 50, 50
xs = np.linspace(0, 11, x_len) # M
ys = np.linspace(0, 30, y_len) # T
z_values = np.array([1, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100])

norm = matplotlib.colors.Normalize(vmin = 0, vmax = 12)
cmap = matplotlib.cm.get_cmap("jet")

for z in z_values:
    xm, ym = np.meshgrid(xs, ys)
    zm = z * np.ones_like(xm)
    sm = 50 * np.ones_like(xm) # Fixing S at a certain value
    rm = 50 * np.ones_like(xm) # Fixing R at a certain value

    r = np.c_[xm.flatten(), zm.flatten(), ym.flatten(), sm.flatten(), rm.flatten()]
    r = pd.DataFrame(r, columns = ['m','c','t','s','r'])
    c = reg_cmp8.predict(r).reshape(x_len, y_len)
    clr = cmap(norm(c))
    surf = ax.plot_surface(xm, ym, zm, rstride=1, cstride=1, linewidth=0.2,
facecolors = clr, shade = False)
```

```
ax.set_xlabel("M (eq.)")
ax.set_ylabel("T (min)")
ax.set_zlabel("C (mM)")
ax.set_xlim(-1,12)
ax.set_ylim(-0.1,31)
ax.view_init(elev = 7, azim =30)
plt.show()
```

Screen-Printed Devices

A crude raw material from (*M*, *T*, *C*, *R*, *S*) = (2, 9, 64, 5, 33) for the screen-printing processes was prepared by performing the reaction in a larger scale as follows. A mixture of bis(1,5-cyclooctadiene)nickel (1.16 g, 4.21 mmol), 2,2'-bipyridyl (658 mg, 4.21 mmol) and 1,5-cyclooctadiene (0.540 ml, 4.21 mmol) in DMF/toluene (5.2 mL/5.2 mL) was stirred at 80 °C for 30 min. To the mixture was added a solution of **1a** (500 mg, 2.00 mmol) and **1b** (21.6 mg, 0.105 mmol) in DMF/toluene (5.2 mL/15.6 mL) dropwise over 9 min by using a syringe pump (YMC YSP-101). The mixture was further stirred at 80 °C for 1 h. After the reaction mixture was cooled down to ambient temperature, 1 M aq. HCl (40 mL) was added, and the mixture was vigorously stirred overnight. The aqueous layer was extracted with $CHCl_3$ (30 mL × 3), and the combined organic layer was washed with brine (70 mL), dried over Na_2SO_4 and concentrated in vacuo. The residue was dissolved in $CHCl_3$ (10 mL) and passed through a pad of silica gel (20 mL), and additional $CHCl_3$ (50 mL) was passed to obtain a crude raw mixture of methylated [*n*]CMPs as a white powder (264 mg). The raw material was dissolved in chlorobenzene at 0.34 wt%, and **3** was also added at 0.06 wt% to furnish the EML solution. In a glove box, the EML solution was drop-casted by pipette on a glass substrate that was precoated by PEDOT:PSS (60 nm)/ITO (120 nm) and covered with a metal mask bearing an image. The substrate was baked at 120 °C for 30 min to include the emitter at 14 wt% in the emission layer. After removing the metal mask, **2** was deposited (60 nm), which was followed by LiF (2 nm) and Al (100 nm) to complete the screen-printed devices (Fig. 5).

Supplementary References

1 A. B. Pangborn, M. A. Giardello, R. H. Grubbs, R. K. Rosen and F. J. Timmers, Safe and Convenient Procedure for Solvent Purification, *Organometallics* 1996, **15**, 1518-1520.

2 G. Taguchi, S. Chowdhury and Y. Wu, *Taguchi's Quality Engineering Handbook*, Wiley, Hoboken, 2005.

3 J. Y. Xue, T. Izumi, A. Yoshii, K. Ikemoto, T, Koretsune, R. Akashi, R, Arita, H. Taka, H. Kita, S. Sota and H. Isobe, Aromatic Hydrocarbon Macrocycles for Highly Efficient Organic Light-Emitting Devices with Single-Layer Architectures, *Chem. Sci.* 2016, **7**, 896-904.

4 J. Y. Xue, K. Ikemoto, N. Takahashi, T. Izumi, H. Taka, H. Kita, S. Sato and H. Isobe, Cyclo-meta-phenylene Revisited: Nickel-Mediated Synthesis, Molecular Structures, and Device Applications, *J. Org. Chem.* 2014, **79**, 9735-9739.

5 (*a*) K. Ikemoto, M. Akiyoshi, T. Mio, K. Nishioka, S. Sato and H. Isobe, Synthesis of a Negatively Curved Nanocarbon Molecule with an Octagonal Omphalos via Design-of-Experiments Optimizations Supplemented by Machine Learning, *Angew. Chem. Int. Ed.* 2022, **61**, e202204035; (*b*) M. Akiyoshi, K. Ikemoto and H. Isobe, Tier-grown Expansion of Design-of-Experiments Parameter Spaces for Synthesis of a Nanometer-scale Macrocycle, *Chem. Asian J.* 2023, **18**, e202201141.

6 F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, Scikit-learn: Machine Learning in Python, *J. Mach. Learn. Res.* 2011, **12**, 2825-2830.

7 McKinney, W. in *Proceedings of the 9th Python in Science Conference*, eds. S. van der Walt and Millman, J. 2010; pp. 56-61.

8 J. D. Hunter, MATPLOTLIB: A 2D Graphics Environment, *Comput. Sci. Eng.* 2007, **9**, 90-95.

9 D. M. Hawkins, S. C. Basak and D. Mills, Assessing Model Fit by Cross-Validation, *J. Chem. Inf. Comput. Sci.* 2003, **43**, 579-586.