Supplementary Material

Quantifying Return on Investment of Differentiating Technologies in Drug Development: A Decision Analytics Framework

Journal of Pharmaceutical Innovation

Ryan P. Nolan[1] and Charles Theuer[1]
[1]Halozyme Therapeutics, San Diego, CA, USA
publications@halozyme.com

Supplementary Table S1: Summary of mean ± standard deviation values for decision tree variables across clinical phases and therapeutic areas.

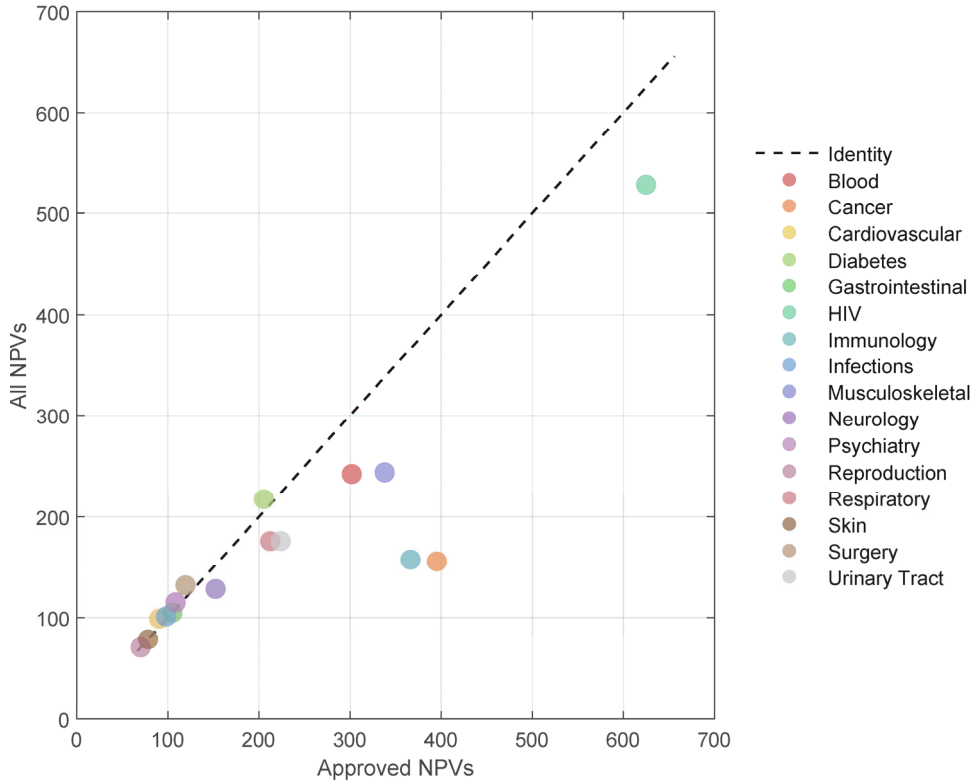
Variable	Phase	Blo	Can	Car	Dia	Gas	HIV	Imm	Inf
Number of drugs*		1316	10847	1929	894	2073	315	1241	5206
POS (%)	**1**	0.81 ± 0.2	0.73 ± 0.15	0.8 ± 0.13	0.65 ± 0.1	0.8 ± 0.17	0.6 ± 0.2	0.69 ± 0.21	0.77 ± 0.19
	2	0.71 ± 0.18	0.41 ± 0.16	0.68 ± 0.17	0.59 ± 0.11	0.65 ± 0.2	0.73 ± 0.17	0.64 ± 0.2	0.73 ± 0.22
	3	0.83 ± 0.21	0.65 ± 0.17	0.76 ± 0.17	0.69 ± 0.18	0.77 ± 0.29	0.9 ± 0.1	0.63 ± 0.24	0.8 ± 0.22
	App	0.94 ± 0.14	0.92 ± 0.13	0.97 ± 0.03	0.91 ± 0.04	0.98 ± 0.04	0.69 ± 0.12	0.88 ± 0.2	0.96 ± 0.11
Duration (months)	**1**	29 ± 24	36 ± 28	25 ± 18	25 ± 22	26 ± 16	31 ± 21	33 ± 24	26 ± 23
	2	39 ± 26	37 ± 28	36 ± 25	34 ± 32	41 ± 31	38 ± 31	37 ± 21	40 ± 34
	3	50 ± 27	38 ± 26	45 ± 24	58 ± 33	50 ± 21	31 ± 17	48 ± 27	41 ± 27
	App	15 ± 9.7	9.6 ± 7.2	17 ± 14	16 ± 13	16 ± 13	11 ± 6.9	14 ± 11	14 ± 13
Cost ($M)	**1**	14 ± 13	28 ± 39	10 ± 10	12 ± 17	10 ± 11	20 ± 25	14 ± 15	6.6 ± 7.7
	2	29 ± 41	58 ± 90	34 ± 47	31 ± 39	24 ± 30	71 ± 95	26 ± 30	34 ± 46
	3	120 ± 190	280 ± 450	280 ± 500	200 ± 390	130 ± 180	550 ± 570	91 ± 120	180 ± 320
Subjects (n)	**1**	40 ± 19	42 ± 14	58 ± 31	61 ± 19	53 ± 27	44 ± 51	38 ± 21	66 ± 64
	2	84 ± 88	84 ± 32	130 ± 81	160 ± 66	110 ± 68	110 ± 96	57 ± 46	280 ± 320
	3	860 ± 1500	520 ± 610	2800 ± 3900	3200 ± 2300	1400 ± 1400	1800 ± 1000	400 ± 460	2200 ± 3500
Cost per Subject ($M)	**1**	0.42 ± 0.44	0.63 ± 0.84	0.18 ± 0.2	0.2 ± 0.27	0.25 ± 0.33	0.52 ± 0.65	0.43 ± 0.44	0.11 ± 0.13
	2	0.5 ± 0.68	0.7 ± 1.1	0.3 ± 0.41	0.23 ± 0.27	0.25 ± 0.34	0.82 ± 1.1	0.48 ± 0.53	0.2 ± 0.32
	3	0.35 ± 0.47	0.56 ± 1.1	0.19 ± 0.36	0.1 ± 0.17	0.12 ± 0.17	0.38 ± 0.35	0.5 ± 0.75	0.13 ± 0.2
Peak Revenue		340 ± 420	370 ± 620	300 ± 520	730 ± 1500	290 ± 510	810 ± 1100	350 ± 500	370 ± 820
Time to Peak Revenue		8.9 ± 5.8	7.6 ± 4.2	5.9 ± 5.6	7.9 ± 5.3	7.7 ± 6.2	5.8 ± 4.5	9.6 ± 6.4	7.8 ± 6
NPV ($M)	**PC**	360 ± 310	180 ± 240	440 ± 370	830 ± 700	400 ± 370	620 ± 590	250 ± 280	650 ± 610
	1	480 ± 400	260 ± 320	570 ± 470	1300 ± 1000	530 ± 470	1100 ± 880	390 ± 400	900 ± 800
	2	720 ± 540	750 ± 700	880 ± 660	2200 ± 1700	860 ± 680	1500 ± 1200	640 ± 560	1400 ± 1100
	3	1000 ± 670	1400 ± 950	1300 ± 830	3400 ± 2300	1300 ± 850	1800 ± 1300	1100 ± 750	1900 ± 1300
	App	1200 ± 760	1600 ± 1100	1400 ± 860	3800 ± 2500	1400 ± 890	2800 ± 1800	1400 ± 890	2200 ± 1400

Abbreviations: **Blo** (Blood), **Can** (Cancer), **Car** (Cardiovascular), **Dia** (Diabetes), **Gas** (Gastrointestinal), **HIV** (HIV), **Imm** (Immunology), **Inf** (Infections), **Mus** (Musculoskeletal), **Neu** (Neurology), **Psy** (Psychiatry), **Rep** (Reproduction), **Res** (Respiratory), **Ski** (Skin), **Sur** (Surgery), **Uri** (Urinary Tract), **$M** (Millions of dollars).
*Total number of drugs in the therapeutic area. For the specific number of drugs used to calculate each variable, refer to the distributions shown in the following figures.

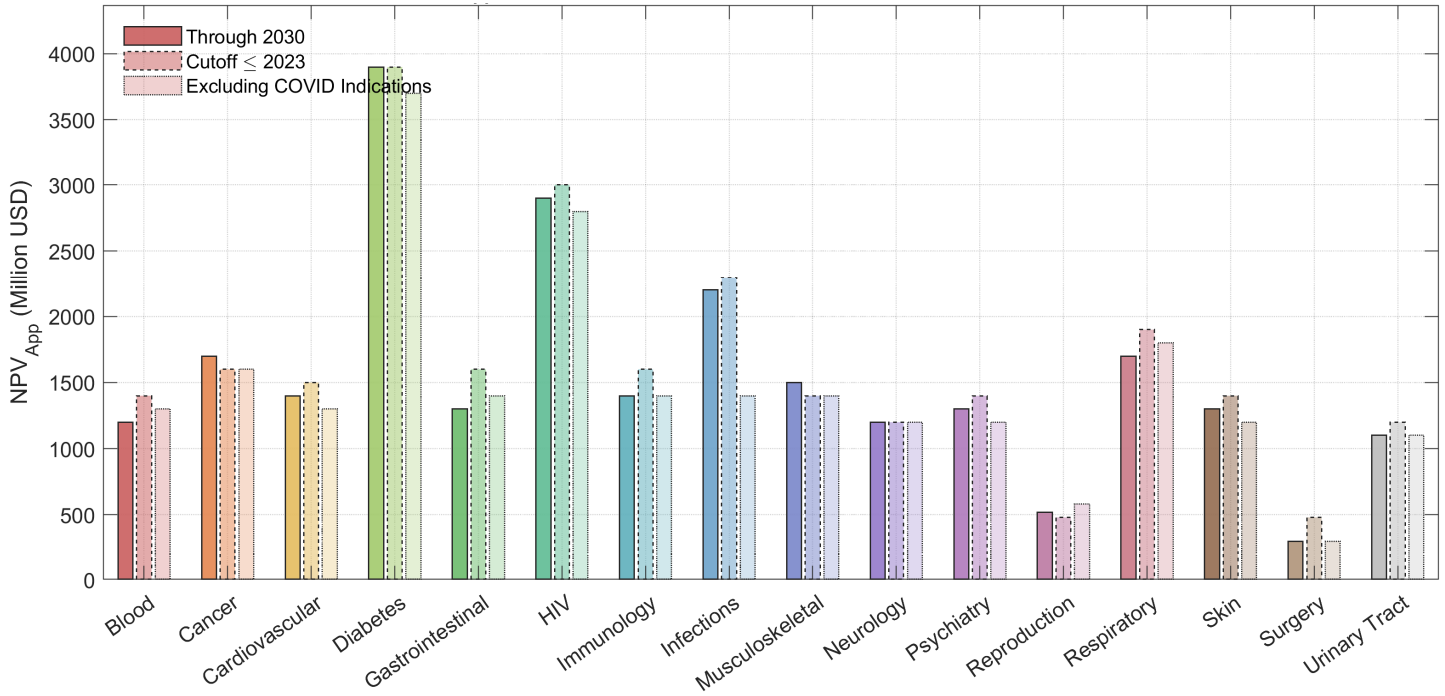
Supplementary Table S1, Continued: Summary of mean ± standard deviation values for decision tree variables across clinical phases and therapeutic areas.

Variable	Phase	Mus	Neu	Psy	Rep	Res	Ski	Sur	Uri
Number of drugs*		1842	3552	934	924	1514	2556	298	521
POS (%)	**1**	0.75 ± 0.16	0.65 ± 0.18	0.63 ± 0.18	0.78 ± 0.18	0.77 ± 0.09	0.81 ± 0.13	0.82 ± 0.18	0.71 ± 0.16
	2	0.66 ± 0.15	0.56 ± 0.16	0.51 ± 0.15	0.68 ± 0.16	0.56 ± 0.11	0.63 ± 0.17	0.67 ± 0.13	0.58 ± 0.25
	3	0.8 ± 0.17	0.62 ± 0.19	0.69 ± 0.21	0.66 ± 0.19	0.61 ± 0.15	0.83 ± 0.16	0.96 ± 0.08	0.67 ± 0.25
	App	0.98 ± 0.03	0.87 ± 0.12	0.96 ± 0.08	0.88 ± 0.19	0.84 ± 0.12	0.96 ± 0.07	0.98 ± 0.05	0.97 ± 0.05
Duration (months)	**1**	30 ± 21	29 ± 20	37 ± 22	27 ± 20	23 ± 17	24 ± 14	25 ± 17	25 ± 17
	2	43 ± 26	42 ± 25	42 ± 30	40 ± 20	46 ± 26	36 ± 23	52 ± 6.5	40 ± 27
	3	42 ± 23	48 ± 26	43 ± 23	56 ± 33	44 ± 25	37 ± 16	46 ± 29	48 ± 28
	App	17 ± 14	16 ± 12	20 ± 14	20 ± 19	18 ± 16	14 ± 8.7	16 ± 10	14 ± 9.5
Cost ($M)	**1**	16 ± 22	11 ± 13	9.1 ± 9.4	6.8 ± 6.7	12 ± 13	11 ± 13	5.4 ± 5.5	5.3 ± 4.3
	2	44 ± 68	26 ± 36	21 ± 29	15 ± 19	43 ± 53	23 ± 31	19 ± 24	25 ± 30
	3	150 ± 250	130 ± 190	160 ± 210	86 ± 130	310 ± 440	120 ± 210	71 ± 98	110 ± 150
Subjects (n)	**1**	44 ± 17	49 ± 23	52 ± 18	46 ± 18	49 ± 11	42 ± 32	50 ± 49	32 ± 16
	2	130 ± 58	100 ± 60	140 ± 59	140 ± 81	120 ± 46	110 ± 69	86 ± 89	75 ± 56
	3	1800 ± 2000	730 ± 740	790 ± 490	1400 ± 1000	1300 ± 1000	1200 ± 850	290 ± 340	700 ± 780
Cost per Subject ($M)	**1**	0.42 ± 0.63	0.23 ± 0.27	0.17 ± 0.17	0.16 ± 0.17	0.26 ± 0.29	0.24 ± 0.29	0.1 ± 0.06	0.17 ± 0.15
	2	0.38 ± 0.57	0.31 ± 0.42	0.15 ± 0.18	0.14 ± 0.16	0.42 ± 0.5	0.21 ± 0.28	0.16 ± 0.17	0.39 ± 0.47
	3	0.19 ± 0.25	0.2 ± 0.3	0.24 ± 0.3	0.15 ± 0.4	0.26 ± 0.31	0.11 ± 0.16	1.2 ± 3	0.29 ± 0.34
Peak Revenue		350 ± 510	270 ± 430	270 ± 460	120 ± 190	370 ± 660	220 ± 460	110 ± 140	310 ± 400
Time to Peak Revenue		8.3 ± 5.3	6.6 ± 4.9	5.8 ± 5.2	5.2 ± 5.4	9 ± 6.4	8.3 ± 6.1	8.9 ± 8.5	4.9 ± 3.6
NPV ($M)	**PC**	460 ± 410	170 ± 190	220 ± 230	110 ± 120	280 ± 290	430 ± 350	130 ± 94	230 ± 230
	1	650 ± 540	280 ± 280	360 ± 350	160 ± 160	370 ± 380	550 ± 450	170 ± 110	330 ± 320
	2	1000 ± 750	530 ± 460	720 ± 590	240 ± 220	700 ± 640	890 ± 660	260 ± 160	600 ± 480
	3	1400 ± 900	940 ± 630	1100 ± 770	420 ± 300	1400 ± 950	1200 ± 780	290 ± 180	1000 ± 620
	App	1500 ± 930	1200 ± 730	1300 ± 820	550 ± 350	1700 ± 1100	1300 ± 830	310 ± 180	1100 ± 650

Abbreviations: **Blo** (Blood), **Can** (Cancer), **Car** (Cardiovascular), **Dia** (Diabetes), **Gas** (Gastrointestinal), **HIV** (HIV), **Imm** (Immunology), **Inf** (Infections), **Mus** (Musculoskeletal), **Neu** (Neurology), **Psy** (Psychiatry), **Rep** (Reproduction), **Res** (Respiratory), **Ski** (Skin), **Sur** (Surgery), **Uri** (Urinary Tract), **$M** (Millions of dollars).
*Total number of drugs in the therapeutic area. For the specific number of drugs used to calculate each variable, refer to the distributions shown in the following figures.

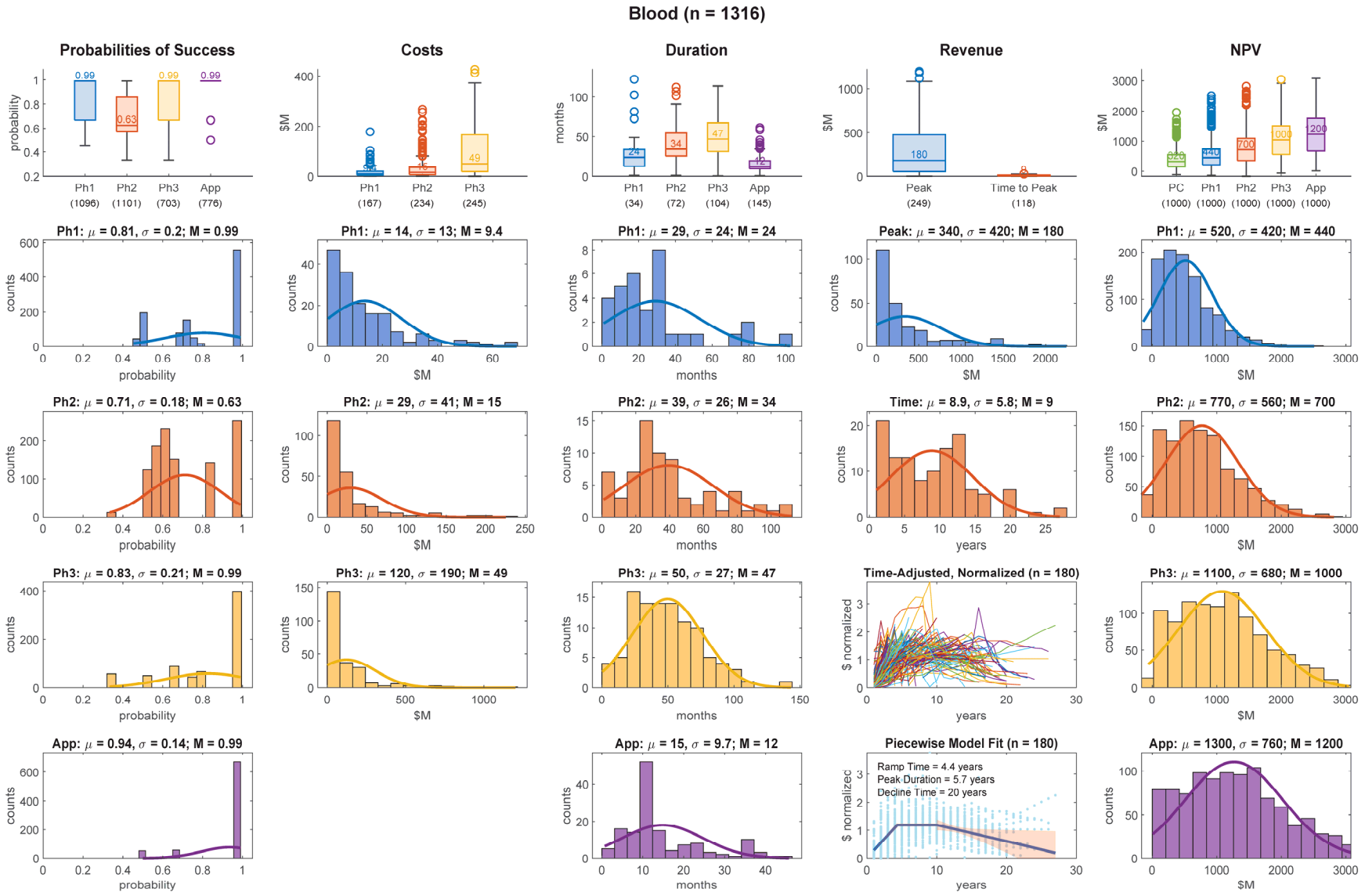


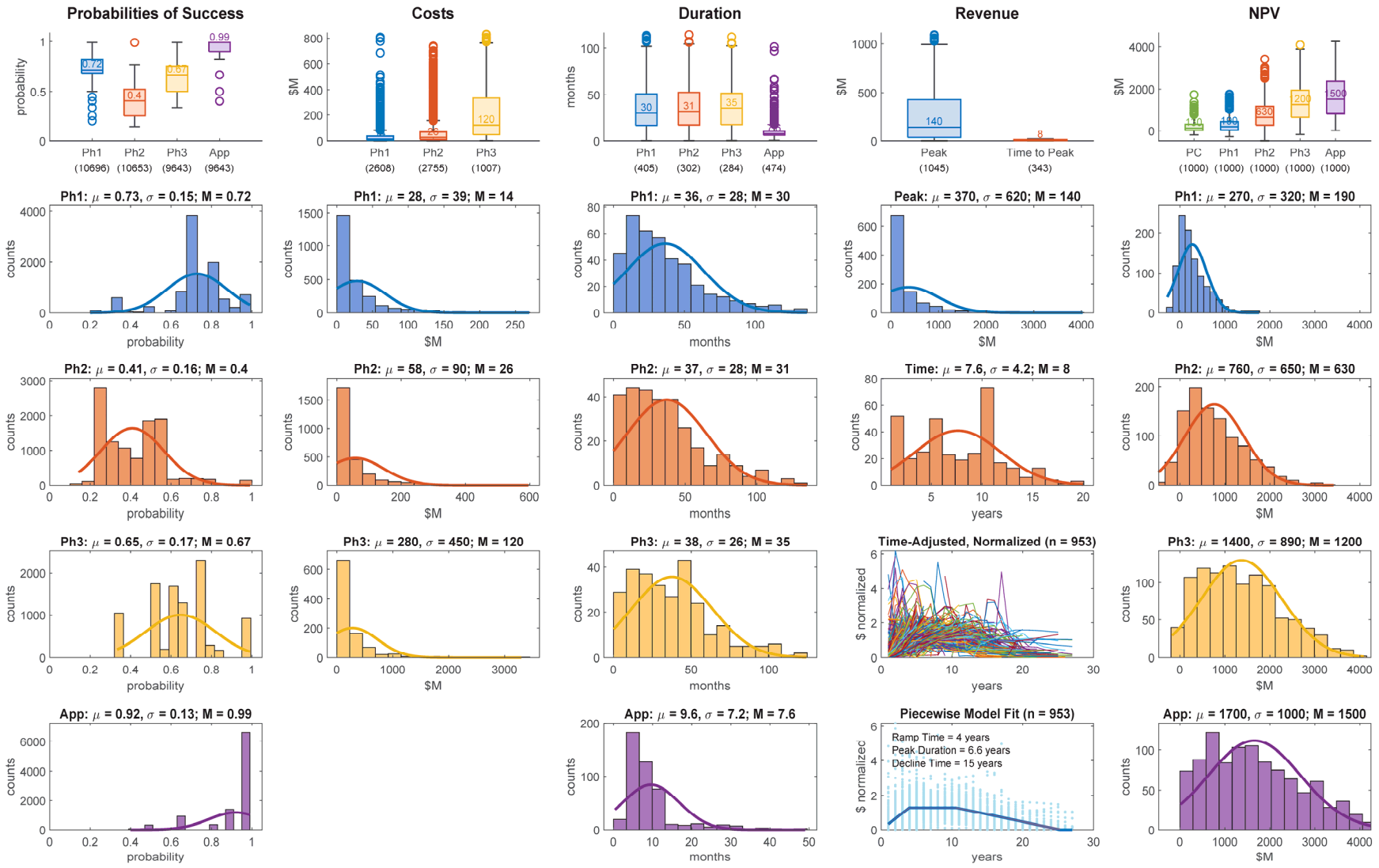
Supplementary Figure S1: External calibration of NPV_{App}. Therapeutic-area comparison of NPV_{App} computed two ways: approved NPV_{App} (x-axis; calculated using only approved/marketed products within each therapeutic area) versus all-dataset NPV_{App} (y-axis; calculated from the full dataset). Points are colored by therapeutic area; the dashed line denotes identity ($y = x$). This face-validity check assesses whether approval-stage values inferred from the full dataset are directionally consistent with values obtained when restricting inputs to approved/marketed products.

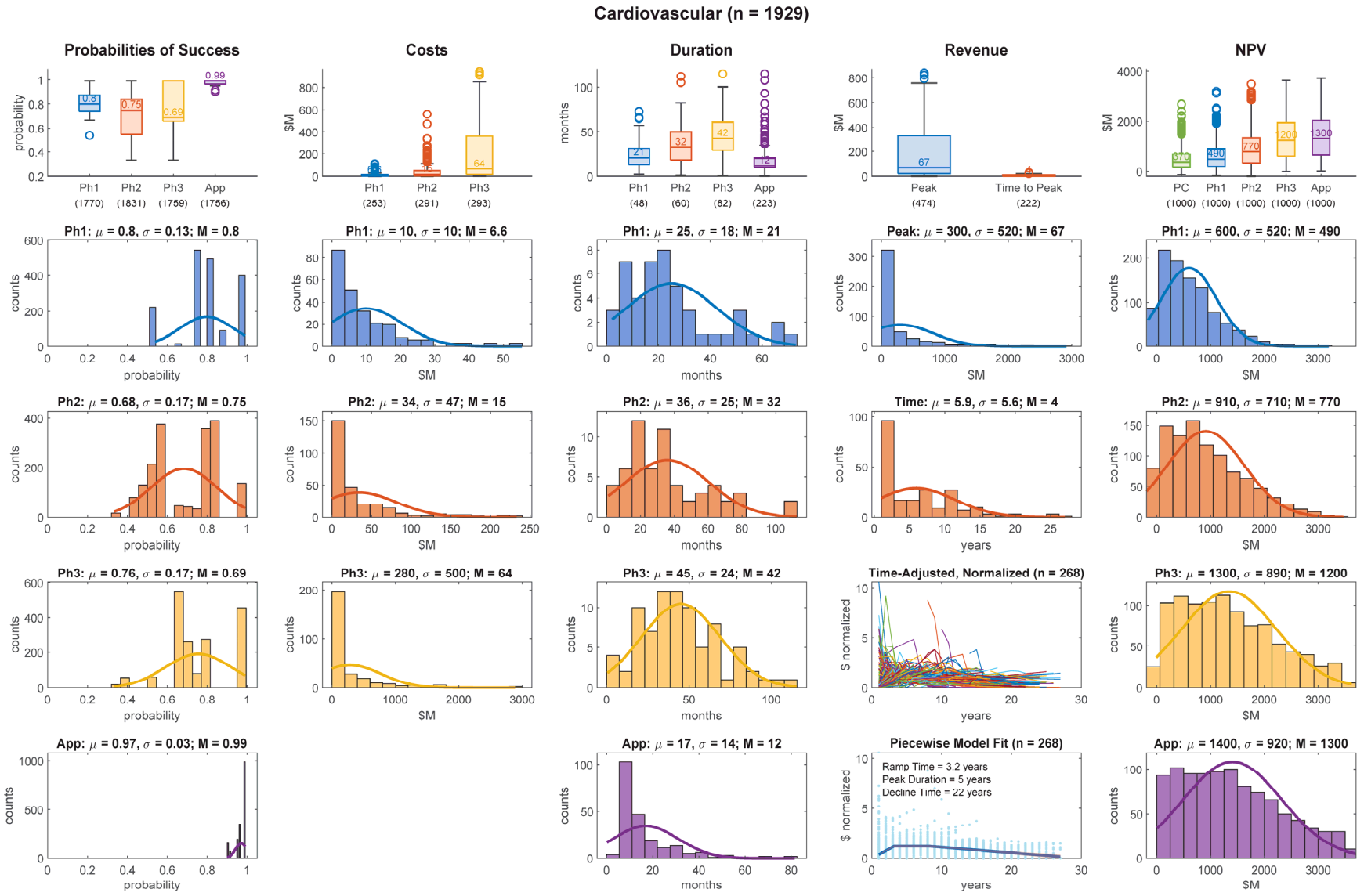


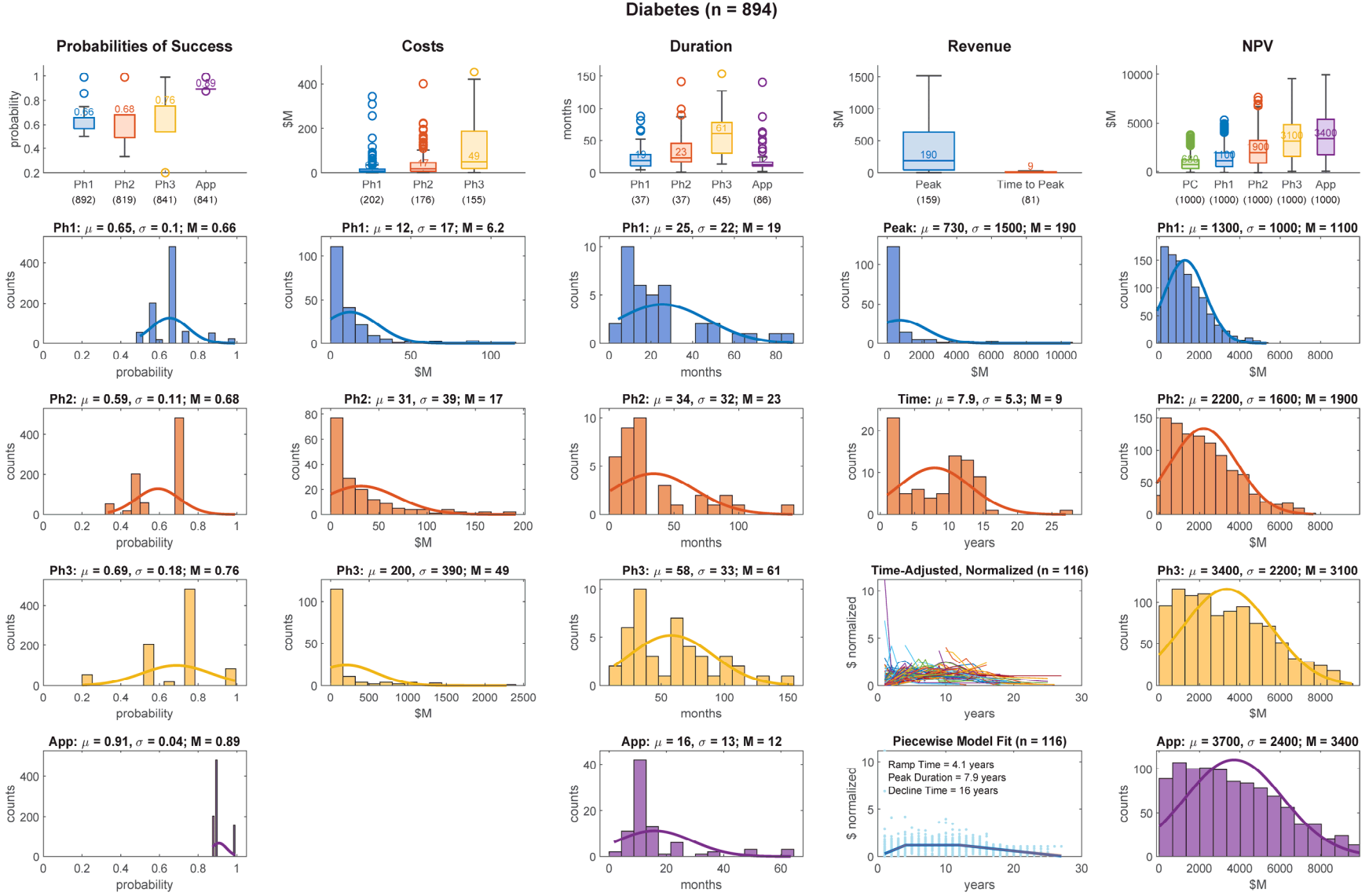
Supplementary Figure S2: NPV_{App} by therapeutic area under horizon and indication sets. Grouped bars show mean NPV_{App} (Million USD) for each therapeutic area under three conditions: Through 2030 (solid fill), Cutoff ≤ 2023 (semi-transparent; dashed edge), and Excluding COVID indications (more transparent; dotted edge). Bars display the same curated dataset and economic assumptions across scenarios; only the revenue horizon (≤ 2023 vs through 2030) or inclusion of COVID-19 indications differs.

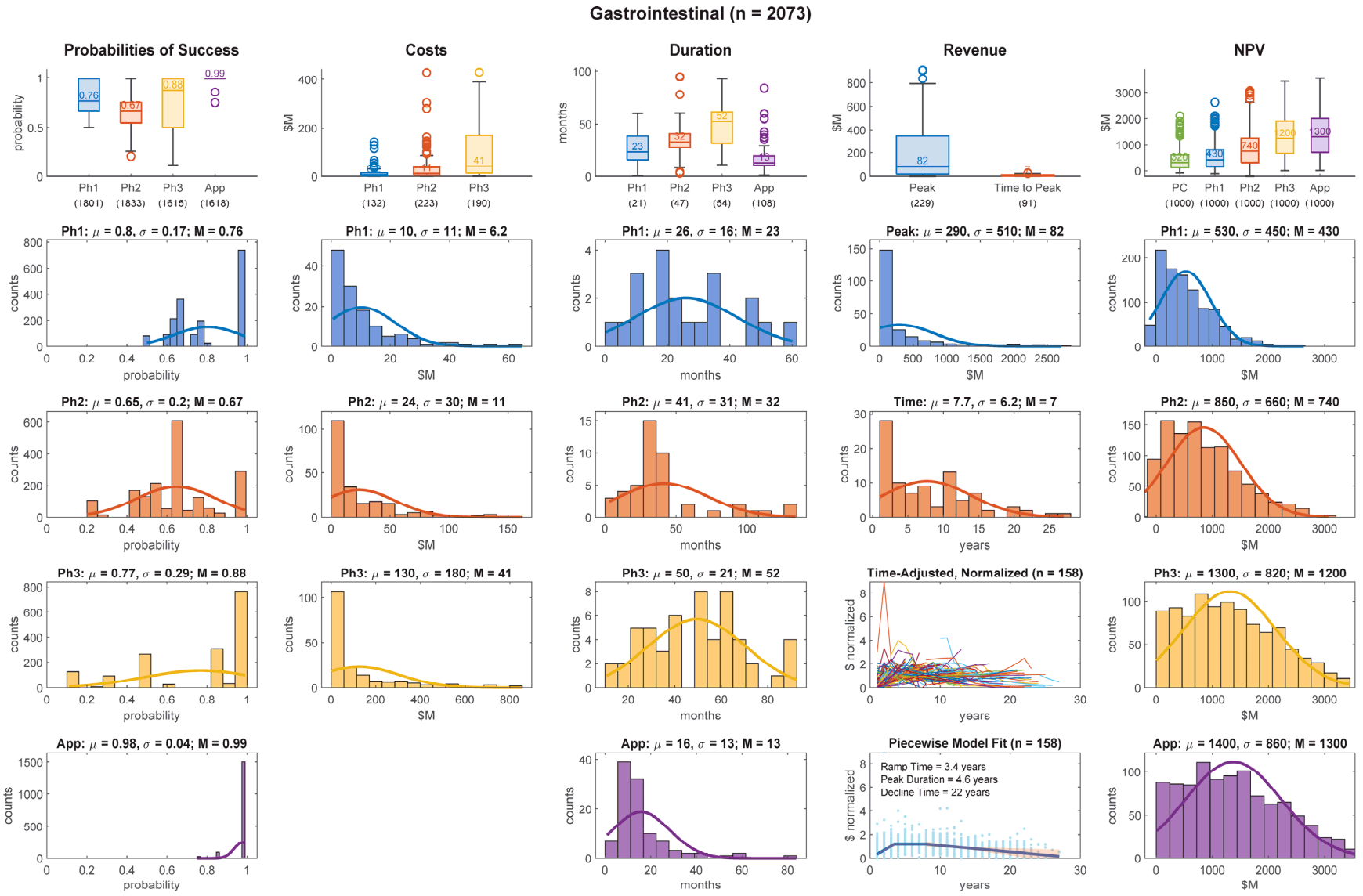
Supplementary Figures S3-S17: Therapeutic area decision tree variable distributions. For the following figures, the top row presents boxplots summarizing the distribution of each variable, showing the median, interquartile range, and whiskers extending to the most extreme non-outlier values. Median values are labeled for clarity, and the number of drugs contributing to each variable is indicated in parentheses. Subsequent rows display the fitted normal distributions for each variable. Each column corresponds to a specific variable. Histogram titles report the mean (μ), standard deviation (σ), and median (M). All monetary values are expressed in millions of dollars ($M).

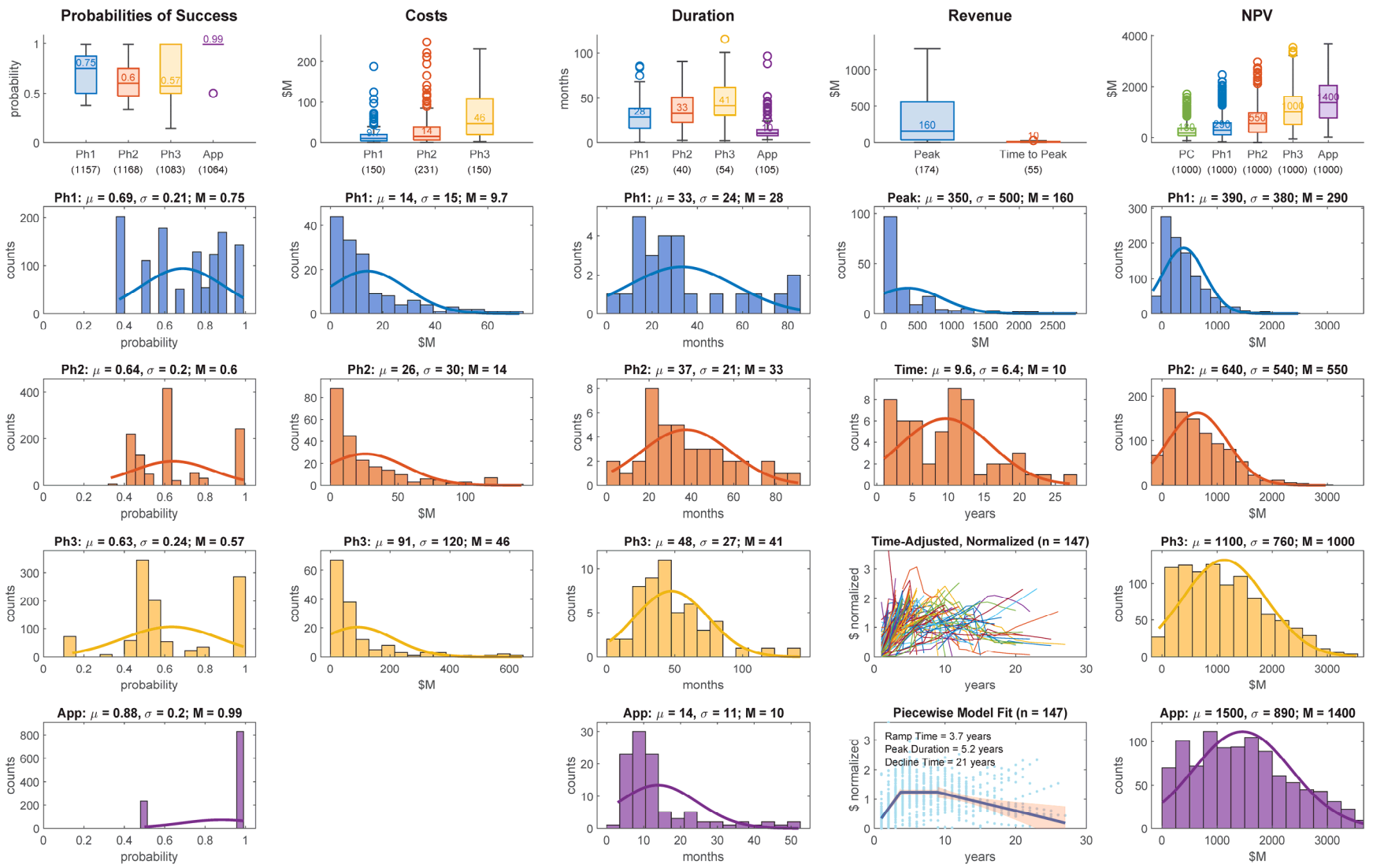

Blood (n = 1316)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (1096)
Ph2 (1101)
Ph3 (703)
App (776)
Ph1 (167)
Ph2 (234)
Ph3 (245)
Ph1 (34)
Ph2 (72)
Ph3 (104)
App (145)
Peak (249)
Time to Peak (118)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
Ph1: μ = 0.81, σ = 0.2; M = 0.99
Ph1: μ = 14, σ = 13; M = 9.4
Ph1: μ = 29, σ = 24; M = 24
Peak: μ = 340, σ = 420; M = 180
Ph1: μ = 520, σ = 420; M = 440
Ph2: μ = 0.71, σ = 0.18; M = 0.63
Ph2: μ = 29, σ = 41; M = 15
Ph2: μ = 39, σ = 26; M = 34
Time: μ = 8.9, σ = 5.8; M = 9
Ph2: μ = 770, σ = 560; M = 700
Ph3: μ = 0.83, σ = 0.21; M = 0.99
Ph3: μ = 120, σ = 190; M = 49
Ph3: μ = 50, σ = 27; M = 47
Time-Adjusted, Normalized (n = 180)
Ph3: μ = 1100, σ = 680; M = 1000
App: μ = 0.94, σ = 0.14; M = 0.99
App: μ = 15, σ = 9.7; M = 12
Piecewise Model Fit (n = 180)
Ramp Time = 4.4 years
Peak Duration = 5.7 years
Decline Time = 20 years
App: μ = 1300, σ = 760; M = 1200
counts
years
$ normalized

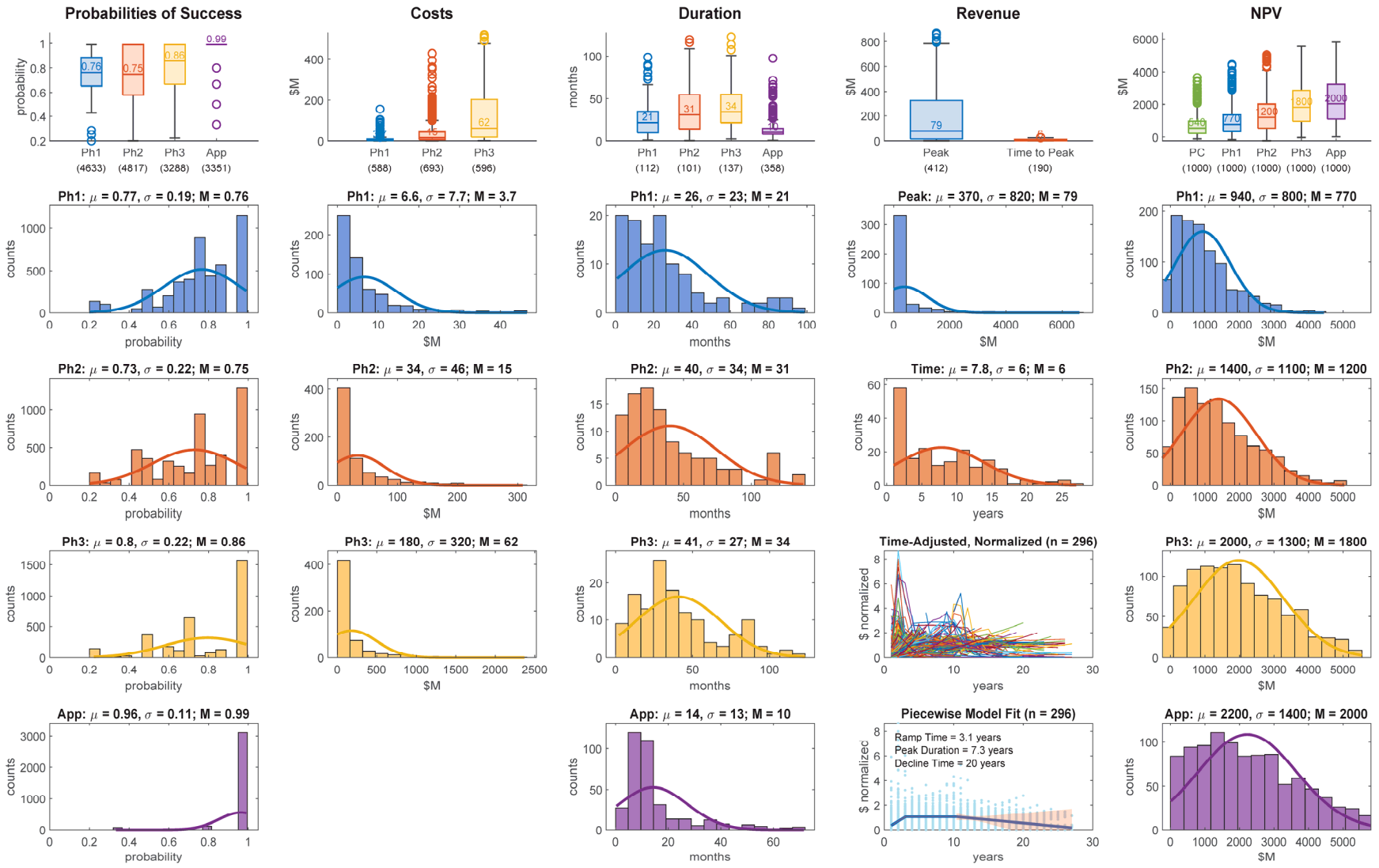

Cancer (n = 10847)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (10696) Ph2 (10653) Ph3 (9643) App (9643)
Ph1 (2608) Ph2 (2755) Ph3 (1007)
Ph1 (405) Ph2 (302) Ph3 (284) App (474)
Peak (1045) Time to Peak (343)
PC (1000) Ph1 (1000) Ph2 (1000) Ph3 (1000) App (1000)
Ph1: μ = 0.73, σ = 0.15; M = 0.72
Ph1: μ = 28, σ = 39; M = 14
Ph1: μ = 36, σ = 28; M = 30
Peak: μ = 370, σ = 620; M = 140
Ph1: μ = 270, σ = 320; M = 190
Ph2: μ = 0.41, σ = 0.16; M = 0.4
Ph2: μ = 58, σ = 90; M = 26
Ph2: μ = 37, σ = 28; M = 31
Time: μ = 7.6, σ = 4.2; M = 8
Ph2: μ = 760, σ = 650; M = 630
Ph3: μ = 0.65, σ = 0.17; M = 0.67
Ph3: μ = 280, σ = 450; M = 120
Ph3: μ = 38, σ = 26; M = 35
Time-Adjusted, Normalized (n = 953)
Ph3: μ = 1400, σ = 890; M = 1200
App: μ = 0.92, σ = 0.13; M = 0.99
App: μ = 9.6, σ = 7.2; M = 7.6
Piecewise Model Fit (n = 953)
Ramp Time = 4 years
Peak Duration = 6.6 years
Decline Time = 15 years
App: μ = 1700, σ = 1000; M = 1500
counts
$ normalized
years

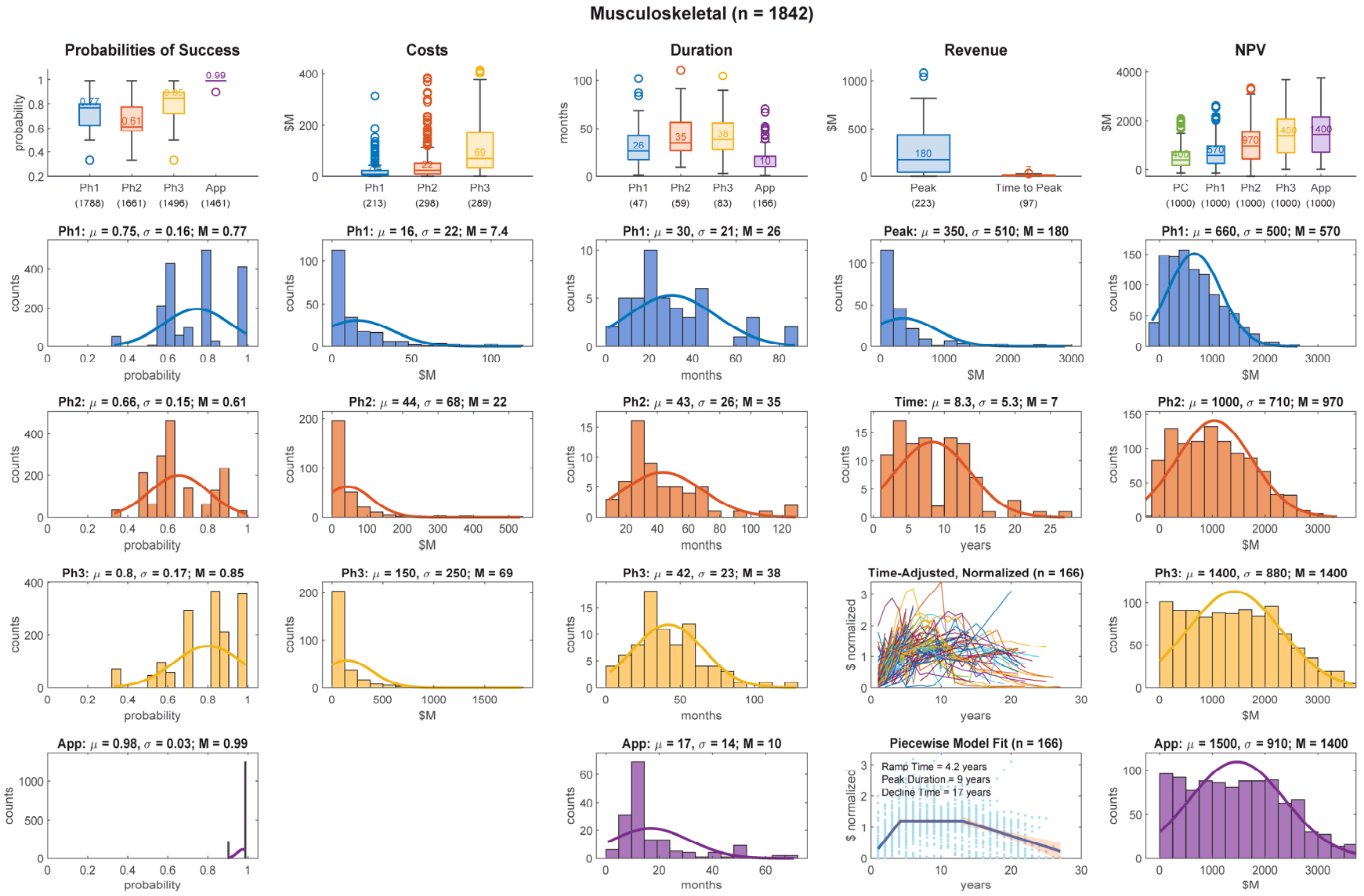

Cardiovascular (n = 1929)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (1770)
Ph2 (1831)
Ph3 (1759)
App (1756)
Ph1 (253)
Ph2 (291)
Ph3 (293)
Ph1 (48)
Ph2 (60)
Ph3 (82)
App (223)
Peak (474)
Time to Peak (222)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
Ph1: μ = 0.8, σ = 0.13; M = 0.8
Ph1: μ = 10, σ = 10; M = 6.6
Ph1: μ = 25, σ = 18; M = 21
Peak: μ = 300, σ = 520; M = 67
Ph1: μ = 600, σ = 520; M = 490
Ph2: μ = 0.68, σ = 0.17; M = 0.75
Ph2: μ = 34, σ = 47; M = 15
Ph2: μ = 36, σ = 25; M = 32
Time: μ = 5.9, σ = 5.6; M = 4
Ph2: μ = 910, σ = 710; M = 770
Ph3: μ = 0.76, σ = 0.17; M = 0.69
Ph3: μ = 280, σ = 500; M = 64
Ph3: μ = 45, σ = 24; M = 42
Time-Adjusted, Normalized (n = 268)
Ph3: μ = 1300, σ = 890; M = 1200
App: μ = 0.97, σ = 0.03; M = 0.99
App: μ = 17, σ = 14; M = 12
Piecewise Model Fit (n = 268)
Ramp Time = 3.2 years
Peak Duration = 5 years
Decline Time = 22 years
App: μ = 1400, σ = 920; M = 1300
counts
$ normalized
years

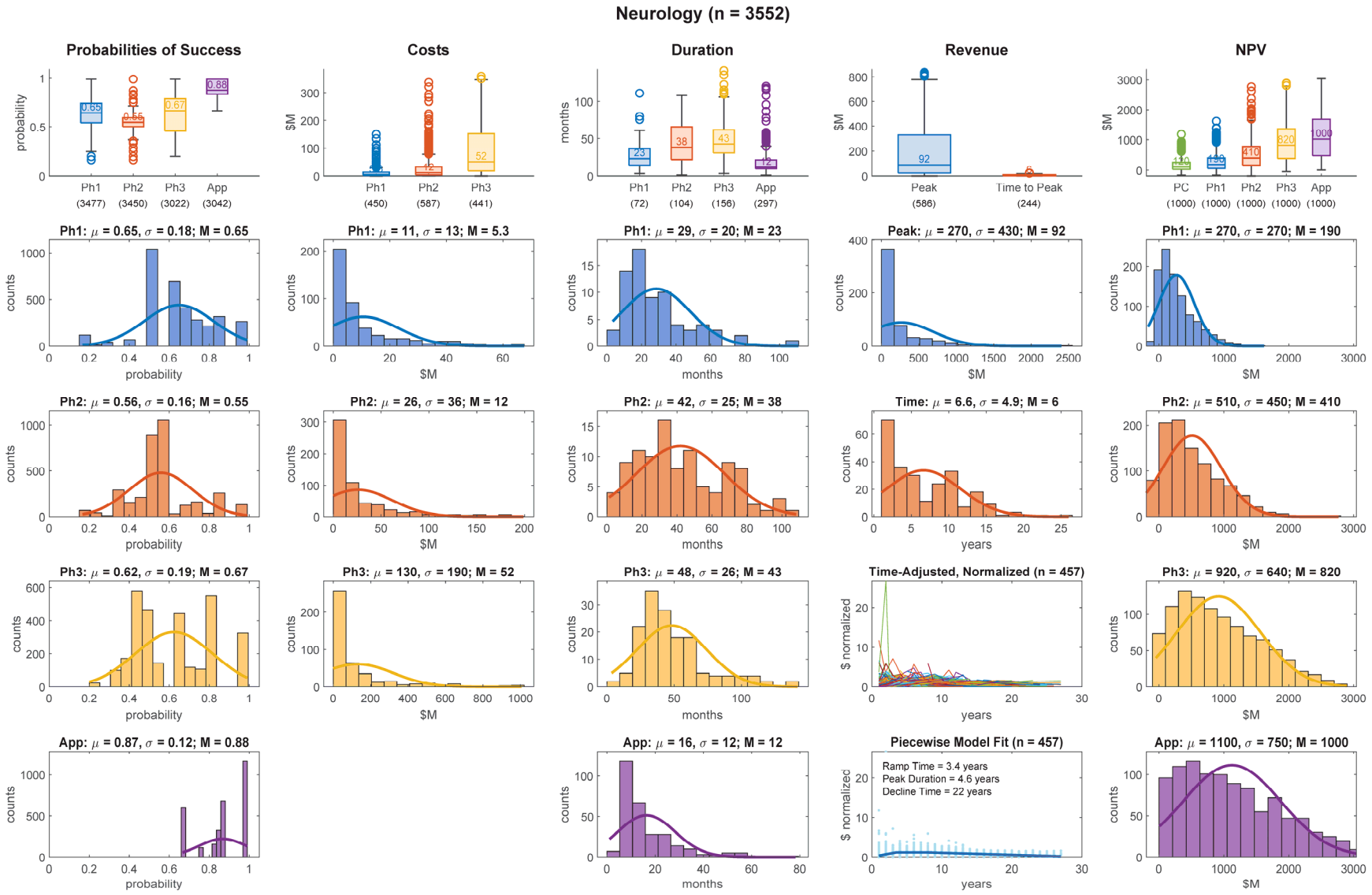

Diabetes (n = 894)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (892)
Ph2 (819)
Ph3 (841)
App (841)
Ph1 (202)
Ph2 (176)
Ph3 (155)
Ph1 (37)
Ph2 (37)
Ph3 (45)
App (86)
Peak (159)
Time to Peak (81)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
0.66
0.68
0.76
0.89
49
23
61
190
9
900
3100
3400
Ph1: μ = 0.65, σ = 0.1; M = 0.66
Ph1: μ = 12, σ = 17; M = 6.2
Ph1: μ = 25, σ = 22; M = 19
Peak: μ = 730, σ = 1500; M = 190
Ph1: μ = 1300, σ = 1000; M = 1100
Ph2: μ = 0.59, σ = 0.11; M = 0.68
Ph2: μ = 31, σ = 39; M = 17
Ph2: μ = 34, σ = 32; M = 23
Time: μ = 7.9, σ = 5.3; M = 9
Ph2: μ = 2200, σ = 1600; M = 1900
Ph3: μ = 0.69, σ = 0.18; M = 0.76
Ph3: μ = 200, σ = 390; M = 49
Ph3: μ = 58, σ = 33; M = 61
Time-Adjusted, Normalized (n = 116)
Ph3: μ = 3400, σ = 2200; M = 3100
App: μ = 0.91, σ = 0.04; M = 0.89
App: μ = 16, σ = 13; M = 12
Piecewise Model Fit (n = 116)
Ramp Time = 4.1 years
Peak Duration = 7.9 years
Decline Time = 16 years
App: μ = 3700, σ = 2400; M = 3400
counts
$ normalized
years

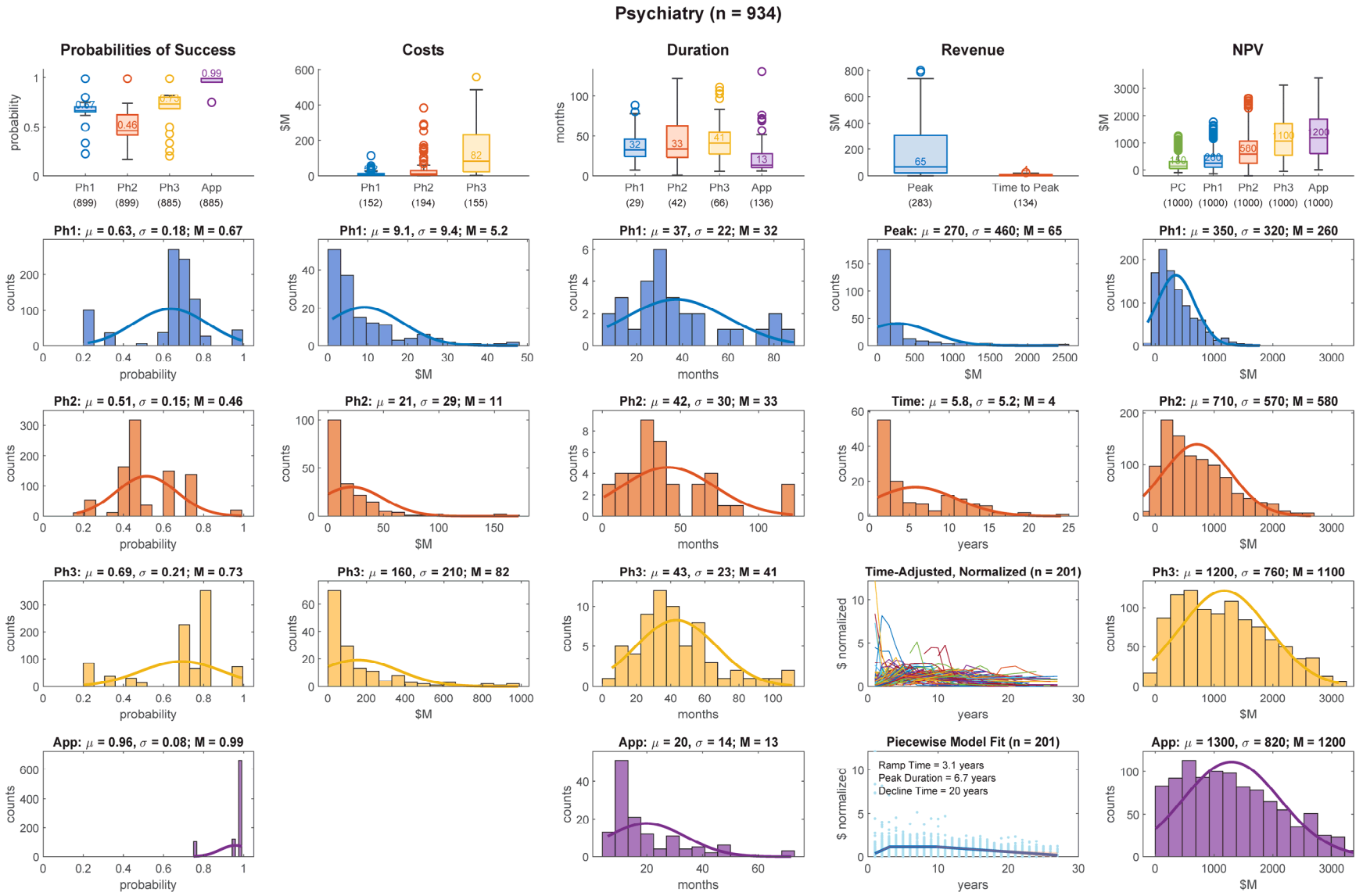

Gastrointestinal (n = 2073)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (1801)
Ph2 (1833)
Ph3 (1615)
App (1618)
Ph1 (132)
Ph2 (223)
Ph3 (190)
Ph1 (21)
Ph2 (47)
Ph3 (54)
App (108)
Peak (229)
Time to Peak (91)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
Ph1: μ = 0.8, σ = 0.17; M = 0.76
Ph1: μ = 10, σ = 11; M = 6.2
Ph1: μ = 26, σ = 16; M = 23
Peak: μ = 290, σ = 510; M = 82
Ph1: μ = 530, σ = 450; M = 430
Ph2: μ = 0.65, σ = 0.2; M = 0.67
Ph2: μ = 24, σ = 30; M = 11
Ph2: μ = 41, σ = 31; M = 32
Time: μ = 7.7, σ = 6.2; M = 7
Ph2: μ = 850, σ = 660; M = 740
Ph3: μ = 0.77, σ = 0.29; M = 0.88
Ph3: μ = 130, σ = 180; M = 41
Ph3: μ = 50, σ = 21; M = 52
Time-Adjusted, Normalized (n = 158)
Ph3: μ = 1300, σ = 820; M = 1200
App: μ = 0.98, σ = 0.04; M = 0.99
App: μ = 16, σ = 13; M = 13
Piecewise Model Fit (n = 158)
Ramp Time = 3.4 years
Peak Duration = 4.6 years
Decline Time = 22 years
App: μ = 1400, σ = 860; M = 1300
counts
$ normalized
years

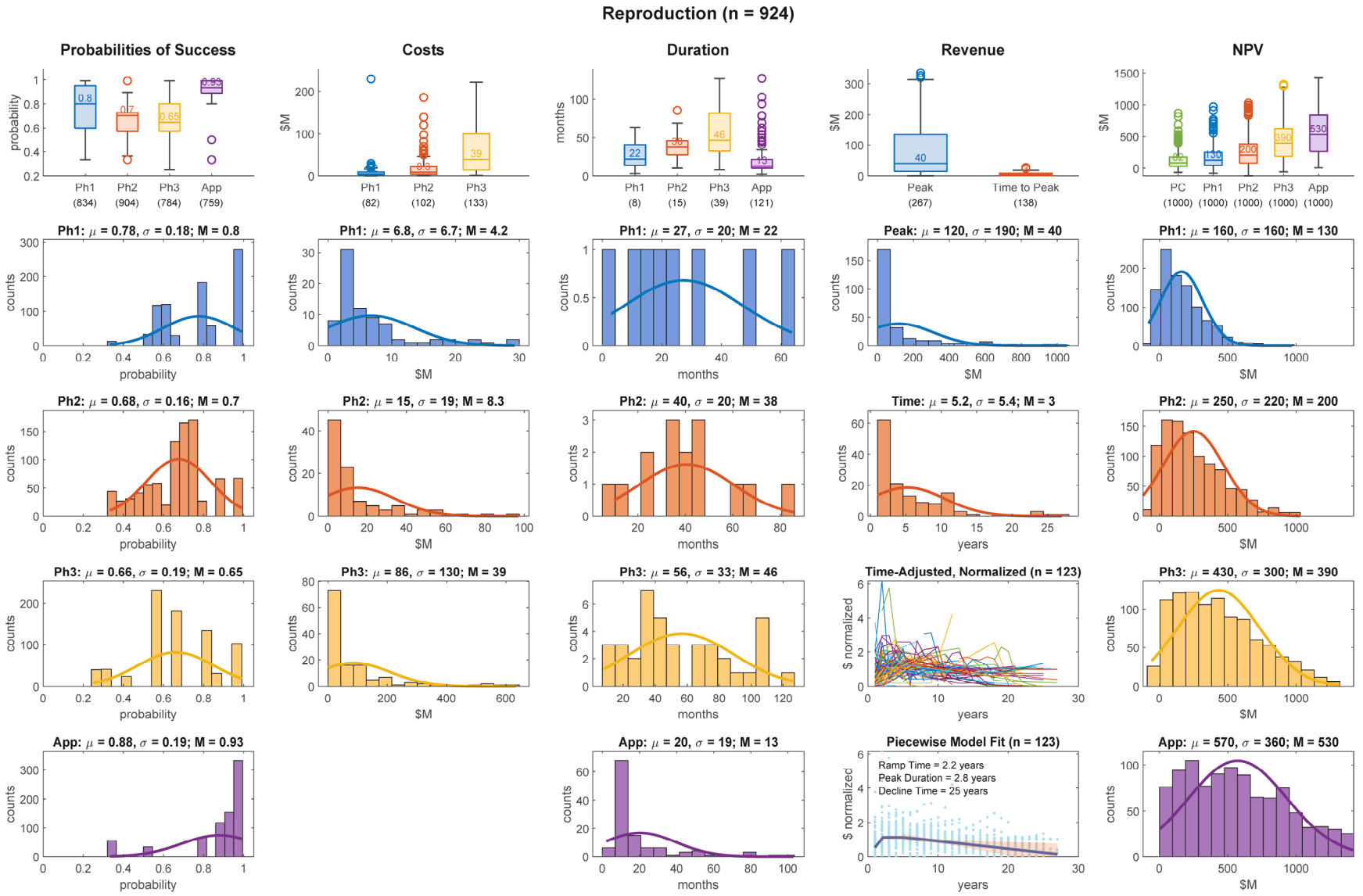

Immunology (n = 1241)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (1157)
Ph2 (1168)
Ph3 (1083)
App (1064)
Ph1 (150)
Ph2 (231)
Ph3 (150)
Ph1 (25)
Ph2 (40)
Ph3 (54)
App (105)
Peak (174)
Time to Peak (55)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
Ph1: μ = 0.69, σ = 0.21; M = 0.75
Ph1: μ = 14, σ = 15; M = 9.7
Ph1: μ = 33, σ = 24; M = 28
Peak: μ = 350, σ = 500; M = 160
Ph1: μ = 390, σ = 380; M = 290
Ph2: μ = 0.64, σ = 0.2; M = 0.6
Ph2: μ = 26, σ = 30; M = 14
Ph2: μ = 37, σ = 21; M = 33
Time: μ = 9.6, σ = 6.4; M = 10
Ph2: μ = 640, σ = 540; M = 550
Ph3: μ = 0.63, σ = 0.24; M = 0.57
Ph3: μ = 91, σ = 120; M = 46
Ph3: μ = 48, σ = 27; M = 41
Time-Adjusted, Normalized (n = 147)
Ph3: μ = 1100, σ = 760; M = 1000
App: μ = 0.88, σ = 0.2; M = 0.99
App: μ = 14, σ = 11; M = 10
Piecewise Model Fit (n = 147)
Ramp Time = 3.7 years
Peak Duration = 5.2 years
Decline Time = 21 years
App: μ = 1500, σ = 890; M = 1400
counts
$ normalized
years

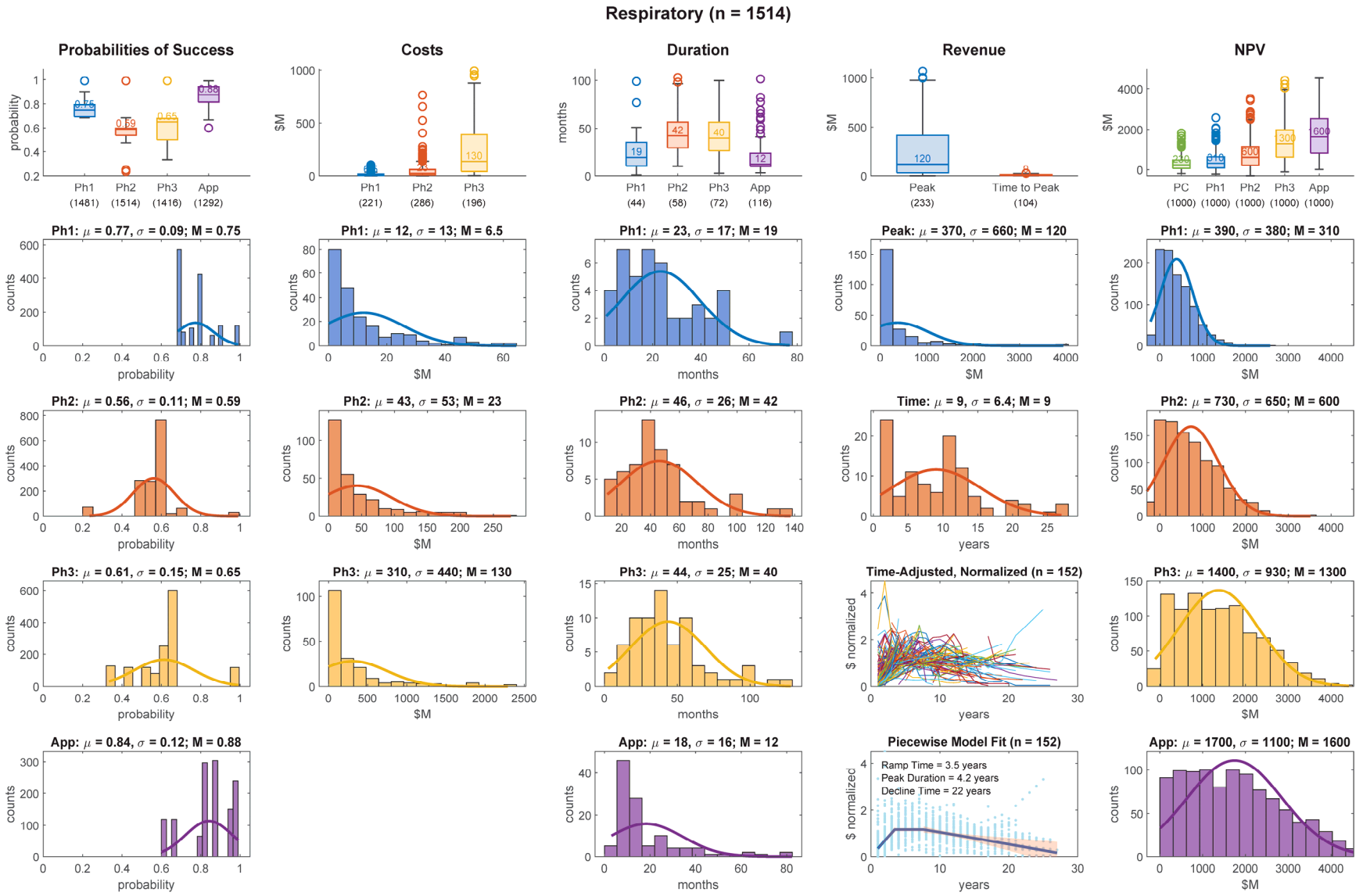

Infections (n = 5206)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (4633)
Ph2 (4817)
Ph3 (3288)
App (3351)
Ph1 (588)
Ph2 (693)
Ph3 (596)
Ph1 (112)
Ph2 (101)
Ph3 (137)
App (358)
Peak (412)
Time to Peak (190)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
Ph1: μ = 0.77, σ = 0.19; M = 0.76
Ph1: μ = 6.6, σ = 7.7; M = 3.7
Ph1: μ = 26, σ = 23; M = 21
Peak: μ = 370, σ = 820; M = 79
Ph1: μ = 940, σ = 800; M = 770
Ph2: μ = 0.73, σ = 0.22; M = 0.75
Ph2: μ = 34, σ = 46; M = 15
Ph2: μ = 40, σ = 34; M = 31
Time: μ = 7.8, σ = 6; M = 6
Ph2: μ = 1400, σ = 1100; M = 1200
Ph3: μ = 0.8, σ = 0.22; M = 0.86
Ph3: μ = 180, σ = 320; M = 62
Ph3: μ = 41, σ = 27; M = 34
Time-Adjusted, Normalized (n = 296)
Ph3: μ = 2000, σ = 1300; M = 1800
App: μ = 0.96, σ = 0.11; M = 0.99
App: μ = 14, σ = 13; M = 10
Piecewise Model Fit (n = 296)
Ramp Time = 3.1 years
Peak Duration = 7.3 years
Decline Time = 20 years
App: μ = 2200, σ = 1400; M = 2000
counts
$ normalized
years

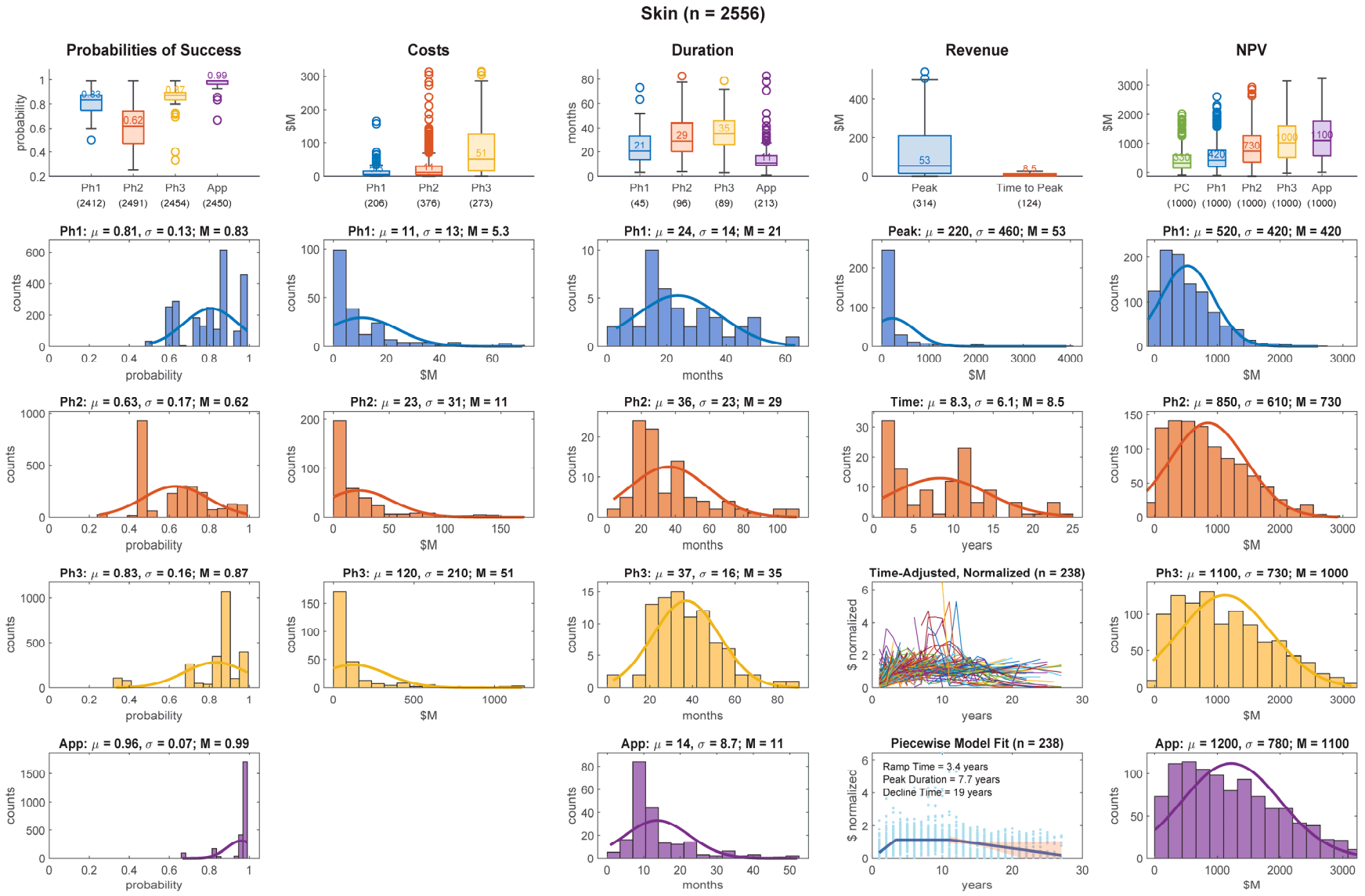

Musculoskeletal (n = 1842)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (1788)
Ph2 (1661)
Ph3 (1496)
App (1461)
Ph1 (213)
Ph2 (298)
Ph3 (289)
Ph1 (47)
Ph2 (59)
Ph3 (83)
App (166)
Peak (223)
Time to Peak (97)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
Ph1: μ = 0.75, σ = 0.16; M = 0.77
Ph1: μ = 16, σ = 22; M = 7.4
Ph1: μ = 30, σ = 21; M = 26
Peak: μ = 350, σ = 510; M = 180
Ph1: μ = 660, σ = 500; M = 570
Ph2: μ = 0.66, σ = 0.15; M = 0.61
Ph2: μ = 44, σ = 68; M = 22
Ph2: μ = 43, σ = 26; M = 35
Time: μ = 8.3, σ = 5.3; M = 7
Ph2: μ = 1000, σ = 710; M = 970
Ph3: μ = 0.8, σ = 0.17; M = 0.85
Ph3: μ = 150, σ = 250; M = 69
Ph3: μ = 42, σ = 23; M = 38
Time-Adjusted, Normalized (n = 166)
Ph3: μ = 1400, σ = 880; M = 1400
App: μ = 0.98, σ = 0.03; M = 0.99
App: μ = 17, σ = 14; M = 10
Piecewise Model Fit (n = 166)
Ramp Time = 4.2 years
Peak Duration = 9 years
Decline Time = 17 years
App: μ = 1500, σ = 910; M = 1400
counts
$ normalized
years

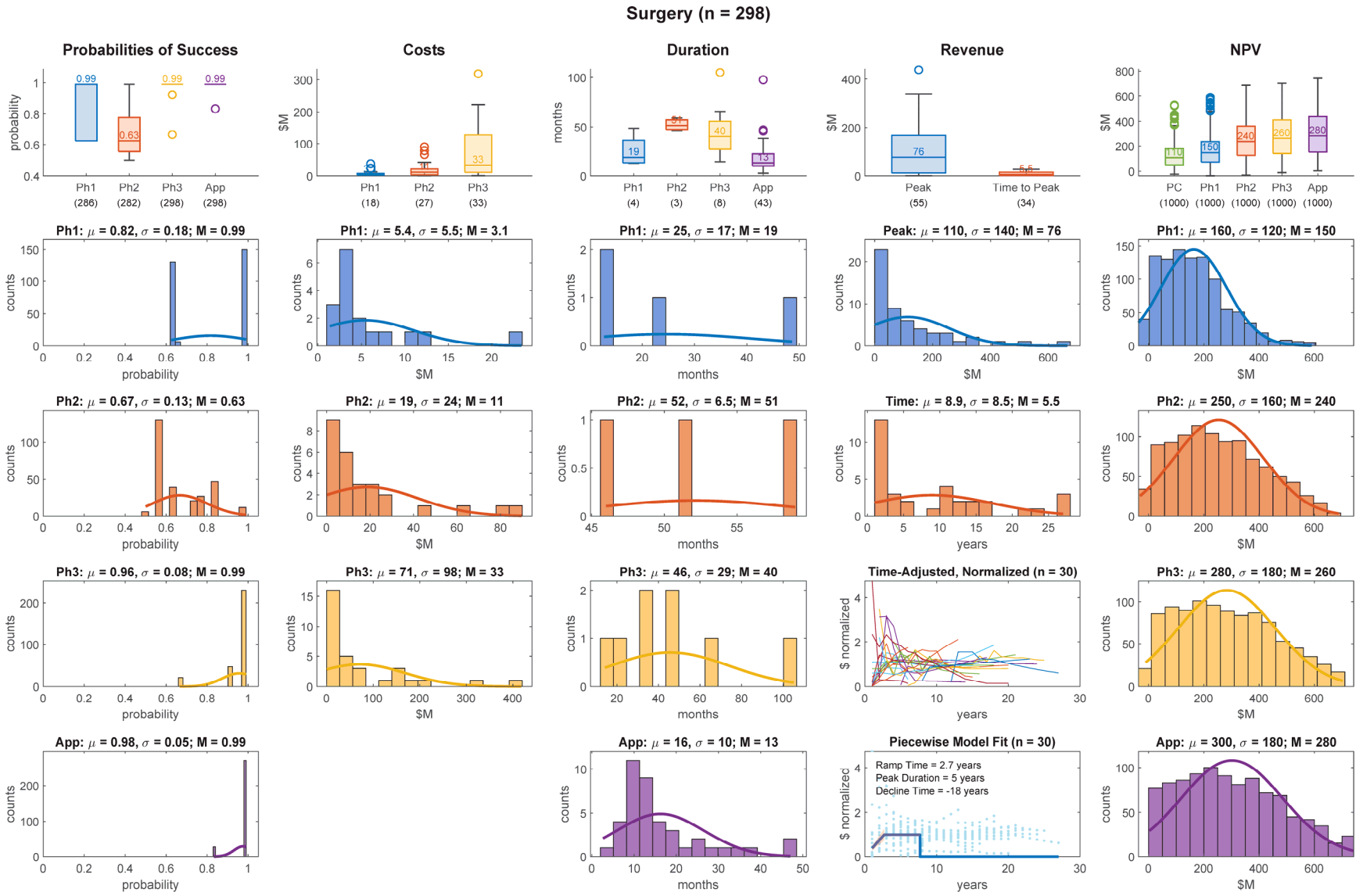

Neurology (n = 3552)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (3477)
Ph2 (3450)
Ph3 (3022)
App (3042)
Ph1 (450)
Ph2 (587)
Ph3 (441)
Ph1 (72)
Ph2 (104)
Ph3 (156)
App (297)
Peak (586)
Time to Peak (244)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
0.65
0.56
0.67
0.88
23
38
43
92
410
820
1000
Ph1: μ = 0.65, σ = 0.18; M = 0.65
Ph1: μ = 11, σ = 13; M = 5.3
Ph1: μ = 29, σ = 20; M = 23
Peak: μ = 270, σ = 430; M = 92
Ph1: μ = 270, σ = 270; M = 190
Ph2: μ = 0.56, σ = 0.16; M = 0.55
Ph2: μ = 26, σ = 36; M = 12
Ph2: μ = 42, σ = 25; M = 38
Time: μ = 6.6, σ = 4.9; M = 6
Ph2: μ = 510, σ = 450; M = 410
Ph3: μ = 0.62, σ = 0.19; M = 0.67
Ph3: μ = 130, σ = 190; M = 52
Ph3: μ = 48, σ = 26; M = 43
Time-Adjusted, Normalized (n = 457)
Ph3: μ = 920, σ = 640; M = 820
App: μ = 0.87, σ = 0.12; M = 0.88
App: μ = 16, σ = 12; M = 12
Piecewise Model Fit (n = 457)
Ramp Time = 3.4 years
Peak Duration = 4.6 years
Decline Time = 22 years
App: μ = 1100, σ = 750; M = 1000
counts
$ normalized
years

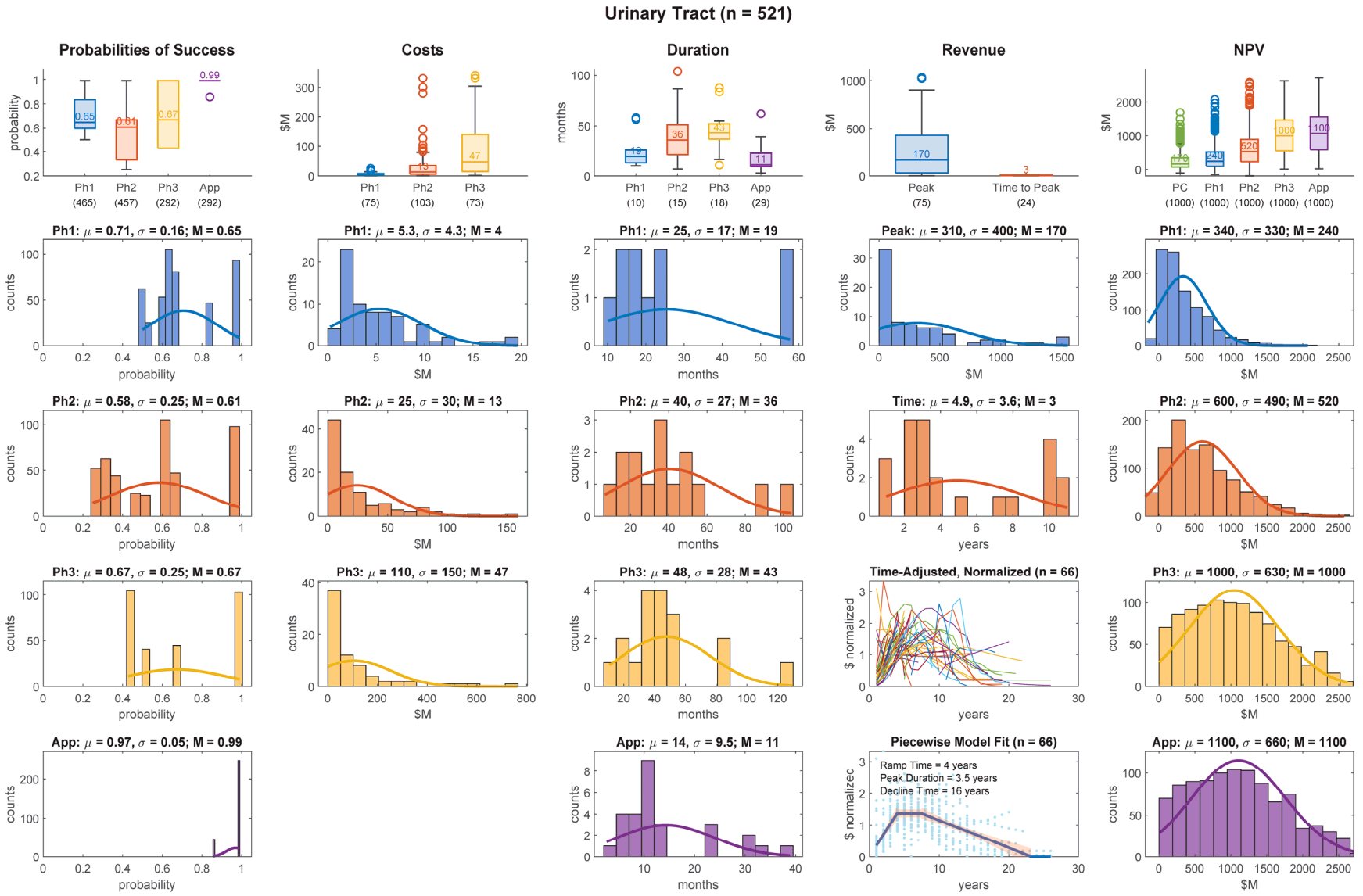

Psychiatry (n = 934)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (899)
Ph2 (899)
Ph3 (885)
App (885)
Ph1 (152)
Ph2 (194)
Ph3 (155)
Ph1 (29)
Ph2 (42)
Ph3 (66)
App (136)
Peak (283)
Time to Peak (134)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
0.99
0.46
0.73
82
32
33
41
13
65
100
200
580
1100
counts
Ph1: μ = 0.63, σ = 0.18; M = 0.67
Ph1: μ = 9.1, σ = 9.4; M = 5.2
Ph1: μ = 37, σ = 22; M = 32
Peak: μ = 270, σ = 460; M = 65
Ph1: μ = 350, σ = 320; M = 260
Ph2: μ = 0.51, σ = 0.15; M = 0.46
Ph2: μ = 21, σ = 29; M = 11
Ph2: μ = 42, σ = 30; M = 33
Time: μ = 5.8, σ = 5.2; M = 4
Ph2: μ = 710, σ = 570; M = 580
Ph3: μ = 0.69, σ = 0.21; M = 0.73
Ph3: μ = 160, σ = 210; M = 82
Ph3: μ = 43, σ = 23; M = 41
Time-Adjusted, Normalized (n = 201)
Ph3: μ = 1200, σ = 760; M = 1100
$ normalized
years
App: μ = 0.96, σ = 0.08; M = 0.99
App: μ = 20, σ = 14; M = 13
Piecewise Model Fit (n = 201)
Ramp Time = 3.1 years
Peak Duration = 6.7 years
Decline Time = 20 years
App: μ = 1300, σ = 820; M = 1200


Reproduction (n = 924)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (834)
Ph2 (904)
Ph3 (784)
App (759)
Ph1 (82)
Ph2 (102)
Ph3 (133)
Ph1 (8)
Ph2 (15)
Ph3 (39)
App (121)
Peak (267)
Time to Peak (138)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
Ph1: μ = 0.78, σ = 0.18; M = 0.8
Ph1: μ = 6.8, σ = 6.7; M = 4.2
Ph1: μ = 27, σ = 20; M = 22
Peak: μ = 120, σ = 190; M = 40
Ph1: μ = 160, σ = 160; M = 130
Ph2: μ = 0.68, σ = 0.16; M = 0.7
Ph2: μ = 15, σ = 19; M = 8.3
Ph2: μ = 40, σ = 20; M = 38
Time: μ = 5.2, σ = 5.4; M = 3
Ph2: μ = 250, σ = 220; M = 200
Ph3: μ = 0.66, σ = 0.19; M = 0.65
Ph3: μ = 86, σ = 130; M = 39
Ph3: μ = 56, σ = 33; M = 46
Time-Adjusted, Normalized (n = 123)
Ph3: μ = 430, σ = 300; M = 390
App: μ = 0.88, σ = 0.19; M = 0.93
App: μ = 20, σ = 19; M = 13
Piecewise Model Fit (n = 123)
Ramp Time = 2.2 years
Peak Duration = 2.8 years
Decline Time = 25 years
App: μ = 570, σ = 360; M = 530
counts
$ normalized
years


Respiratory (n = 1514)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (1481)
Ph2 (1514)
Ph3 (1416)
App (1292)
Ph1 (221)
Ph2 (286)
Ph3 (196)
Ph1 (44)
Ph2 (58)
Ph3 (72)
App (116)
Peak (233)
Time to Peak (104)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
Ph1: μ = 0.77, σ = 0.09; M = 0.75
Ph1: μ = 12, σ = 13; M = 6.5
Ph1: μ = 23, σ = 17; M = 19
Peak: μ = 370, σ = 660; M = 120
Ph1: μ = 390, σ = 380; M = 310
Ph2: μ = 0.56, σ = 0.11; M = 0.59
Ph2: μ = 43, σ = 53; M = 23
Ph2: μ = 46, σ = 26; M = 42
Time: μ = 9, σ = 6.4; M = 9
Ph2: μ = 730, σ = 650; M = 600
Ph3: μ = 0.61, σ = 0.15; M = 0.65
Ph3: μ = 310, σ = 440; M = 130
Ph3: μ = 44, σ = 25; M = 40
Time-Adjusted, Normalized (n = 152)
Ph3: μ = 1400, σ = 930; M = 1300
App: μ = 0.84, σ = 0.12; M = 0.88
App: μ = 18, σ = 16; M = 12
Piecewise Model Fit (n = 152)
Ramp Time = 3.5 years
Peak Duration = 4.2 years
Decline Time = 22 years
App: μ = 1700, σ = 1100; M = 1600
counts
$ normalized
years


Skin (n = 2556)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (2412)
Ph2 (2491)
Ph3 (2454)
App (2450)
Ph1 (206)
Ph2 (376)
Ph3 (273)
Ph1 (45)
Ph2 (96)
Ph3 (89)
App (213)
Peak (314)
Time to Peak (124)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
Ph1: μ = 0.81, σ = 0.13; M = 0.83
Ph1: μ = 11, σ = 13; M = 5.3
Ph1: μ = 24, σ = 14; M = 21
Peak: μ = 220, σ = 460; M = 53
Ph1: μ = 520, σ = 420; M = 420
Ph2: μ = 0.63, σ = 0.17; M = 0.62
Ph2: μ = 23, σ = 31; M = 11
Ph2: μ = 36, σ = 23; M = 29
Time: μ = 8.3, σ = 6.1; M = 8.5
Ph2: μ = 850, σ = 610; M = 730
Ph3: μ = 0.83, σ = 0.16; M = 0.87
Ph3: μ = 120, σ = 210; M = 51
Ph3: μ = 37, σ = 16; M = 35
Time-Adjusted, Normalized (n = 238)
Ph3: μ = 1100, σ = 730; M = 1000
App: μ = 0.96, σ = 0.07; M = 0.99
App: μ = 14, σ = 8.7; M = 11
Piecewise Model Fit (n = 238)
Ramp Time = 3.4 years
Peak Duration = 7.7 years
Decline Time = 19 years
App: μ = 1200, σ = 780; M = 1100
counts
$ normalized
years


Surgery (n = 298)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (286)
Ph2 (282)
Ph3 (298)
App (298)
Ph1 (18)
Ph2 (27)
Ph3 (33)
Ph1 (4)
Ph2 (3)
Ph3 (8)
App (43)
Peak (55)
Time to Peak (34)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
Ph1: μ = 0.82, σ = 0.18; M = 0.99
Ph1: μ = 5.4, σ = 5.5; M = 3.1
Ph1: μ = 25, σ = 17; M = 19
Peak: μ = 110, σ = 140; M = 76
Ph1: μ = 160, σ = 120; M = 150
Ph2: μ = 0.67, σ = 0.13; M = 0.63
Ph2: μ = 19, σ = 24; M = 11
Ph2: μ = 52, σ = 6.5; M = 51
Time: μ = 8.9, σ = 8.5; M = 5.5
Ph2: μ = 250, σ = 160; M = 240
Ph3: μ = 0.96, σ = 0.08; M = 0.99
Ph3: μ = 71, σ = 98; M = 33
Ph3: μ = 46, σ = 29; M = 40
Time-Adjusted, Normalized (n = 30)
Ph3: μ = 280, σ = 180; M = 260
App: μ = 0.98, σ = 0.05; M = 0.99
App: μ = 16, σ = 10; M = 13
Piecewise Model Fit (n = 30)
Ramp Time = 2.7 years
Peak Duration = 5 years
Decline Time = -18 years
App: μ = 300, σ = 180; M = 280
counts
years
$ normalized


Urinary Tract (n = 521)
Probabilities of Success
Costs
Duration
Revenue
NPV
probability
$M
months
Ph1 (465)
Ph2 (457)
Ph3 (292)
App (292)
Ph1 (75)
Ph2 (103)
Ph3 (73)
Ph1 (10)
Ph2 (15)
Ph3 (18)
App (29)
Peak (75)
Time to Peak (24)
PC (1000)
Ph1 (1000)
Ph2 (1000)
Ph3 (1000)
App (1000)
Ph1: μ = 0.71, σ = 0.16; M = 0.65
Ph1: μ = 5.3, σ = 4.3; M = 4
Ph1: μ = 25, σ = 17; M = 19
Peak: μ = 310, σ = 400; M = 170
Ph1: μ = 340, σ = 330; M = 240
Ph2: μ = 0.58, σ = 0.25; M = 0.61
Ph2: μ = 25, σ = 30; M = 13
Ph2: μ = 40, σ = 27; M = 36
Time: μ = 4.9, σ = 3.6; M = 3
Ph2: μ = 600, σ = 490; M = 520
Ph3: μ = 0.67, σ = 0.25; M = 0.67
Ph3: μ = 110, σ = 150; M = 47
Ph3: μ = 48, σ = 28; M = 43
Time-Adjusted, Normalized (n = 66)
Ph3: μ = 1000, σ = 630; M = 1000
App: μ = 0.97, σ = 0.05; M = 0.99
App: μ = 14, σ = 9.5; M = 11
Piecewise Model Fit (n = 66)
Ramp Time = 4 years
Peak Duration = 3.5 years
Decline Time = 16 years
App: μ = 1100, σ = 660; M = 1100
counts
years
$ normalized

Supplementary Code S1: MATLAB. Scripts implementing the full decision-analytics workflow, including database curation and harmonization, therapeutic-area mapping, estimation of phase-specific probabilities, costs, durations, subject counts, revenue modeling, NPV calculations, sensitivity analyses, and generation of all tables and figures.

```
function create_wkspc

%% NOTES
% Database Curation
% Two Excel-based datasets were imported:
% (1) containing product-level clinical development data
% (2) containing sales information.
% The primary dataset was curated through several preprocessing steps to ensure consistency
and analytical readiness:
%
% Variable Renaming:
% - Column headers were renamed to simplified, standardized labels to improve readability and
facilitate downstream analysis.
% - Only relevant variables were retained, and extraneous columns were removed to streamline
the dataset.
%
% Therapeutic Area Standardizing:
% - Therapeutic area (TA) entries were mapped to a controlled vocabulary using a predefined
key-value mapping.
% This ensured consistent categorization across similar or synonymous TA labels (e.g.,
“Cardiovascular Diseases” was standardized to “Cardiovascular”).
%
% Row Filtering:
% - Rows with missing or undefined therapeutic area labels were excluded.
% - Rows were removed if all key numerical fields (e.g., phase transition probabilities, trial
durations, costs, subject counts, and revenue metrics) were empty or zero-valued.
% - Duplicate entries were identified and removed based on a subset of key variables, using
standardized placeholders for missing values to ensure accurate comparison.
%
% Workspace Finalization:
% - The curated dataset was saved for subsequent analysis.

%% Database Curation
clear all; clc; close all;
%
% database_filename = 'database 04152025.xlsx';
% product_sales_filename = 'product_sales_evaluate.xlsx';
%
% % Read data from Excel file with the specified options
% A = readtable(database_filename, 'Sheet', 'Product Indication level', 'ReadRowNames', false,
'PreserveVariableNames', false);
% % Read data from Excel file
% S = readtable(product_sales_filename, 'Sheet', 'Report', 'ReadRowNames', false,
'PreserveVariableNames', false);
%
% save('wkspc_raw','A','S');
load wkspc_raw A S

%% Variable Renaming
% Rename variables to simpler names
variableNames = {
    'ev_ww_pi__pii__indication_l3_nda', 'POS_Ph1';
    'ev_ww_pii__piii__indication_l3_nda', 'POS_Ph2';
    'ev_usa_piii__filed__indication_l3_nda', 'POS_Ph3';
    'ev_usa_filed__approved__indication_l3_nda', 'POS_App';
    'ev_product__indication_total_registered_trial_cost_pi_m', 'Cost_Ph1';
    'ev_product__indication_total_registered_trial_cost_pii_m', 'Cost_Ph2';
```

```
    'ev_product__indication_total_registered_trial_cost_piii_m', 'Cost_Ph3';
    'ev_time_in_pi_months_actual', 'Duration_Ph1';
    'ev_time_in_pii_months_actual', 'Duration_Ph2';
    'ev_time_in_piii_months_actual', 'Duration_Ph3';
    'ev_filing_time_months_actual', 'Duration_App';
    'ev_pi_patient_enrolment_per_product_median_benchmark_indication', 'Subjects_Ph1';
    'ev_pii_patient_enrolment_per_product_median_benchmark_indicatio', 'Subjects_Ph2';
    'ev_piii_patient_enrolment_per_product_median_benchmark_indicati', 'Subjects_Ph3';
    'ev_usa_peak_sales_benchmark_time_adjusted_indication_l3_nda_m', 'Revenue_Peak';
    'ev_usa_median_time_to_peak_benchmark_years_indication_l3_nda', 'Revenue_Time';
    'company','Company';
    'product_name','Product';
    'generic_name','Generic';
    'molecule_type','Modality';
    'route','Route';
    'disease_area', 'TA';
    'indication', 'Indication';
    'ev_ww_indication_status_current', 'Status';
    'product_id','ID';};

% Redefine headers in A to preferred names
for i = 1:size(variableNames, 1)
    A.Properties.VariableNames{variableNames{i, 1}} = variableNames{i, 2};
end

% Find the indices of the specified columns in A
[~, colIndices] = ismember(variableNames(:,2), A.Properties.VariableNames);
A = A(:, colIndices);
for i = 1:16
    if iscell(A{:, i})
        A.(A.Properties.VariableNames{i}) = str2double(A{:, i}); % convert cells to double
    end
end

TA_list = unique(A.TA);
TA_list(strcmp(TA_list, '')) = [];

save wkspc_1 A S

%% Therapeutic Area Standardizing
% Define mapping as key-value pairs
taPairs = {
    'Blood', 'Blood';
    'Blood Disorders', 'Blood';
    'Cancer', 'Cancer';
    'Cardiovascular', 'Cardiovascular';
    'Cardiovascular Diseases', 'Cardiovascular';
    'Diabetes', 'Diabetes';
    'Gastro-intestinal', 'Gastrointestinal';
    'Gastrointestinal Diseases', 'Gastrointestinal';
    'HIV & related conditions', 'HIV';
    'Hepatic & biliary', 'Hepatic & Biliary';
    'Hormone', 'Hormone';
    'Endocrine/Metabolism', 'Hormone';
    'Immunology', 'Immunology';
    'Inflammation/Immune', 'Immunology';
    'Antisynthetase Syndrome', 'Immunology';
    'Infections', 'Infections';
    'Miscellaneous', 'Miscellaneous';
    'Other Therapeutic Areas', 'Miscellaneous';
    'Musculoskeletal', 'Musculoskeletal';
```

```
    'Musculoskeletal Diseases', 'Musculoskeletal';
    'Pain Management', 'Musculoskeletal';
    'Neurology', 'Neurology';
    'CNS', 'Neurology';
    'Psychiatry', 'Psychiatry';
    'Reproduction', 'Reproduction';
    'Reproductive Health', 'Reproduction';
    'Respiratory', 'Respiratory';
    'Sensory organs', 'Sensory organs';
    'Eye Diseases', 'Sensory organs';
    'Skin', 'Skin';
    'Skin Disorders', 'Skin';
    'Surgery', 'Surgery';
    'Urinary tract', 'Urinary Tract';
    'Genitourinary Diseases', 'Urinary Tract'};

% Create map
taMap = containers.Map(taPairs(:,1), taPairs(:,2));

% Initialize logical index for rows to keep
rowsToKeep = true(height(A), 1);

% Apply mapping and mark rows with empty TA for deletion
for i = 1:height(A)
    disp(i)
    ta = A.TA{i};
    if isempty(ta) || (isstring(ta) && ta == "")
        rowsToKeep(i) = false;  % Mark for deletion
    elseif isKey(taMap, ta)
        A.TA{i} = taMap(ta);    % Map to standardized category
    end
end

% Delete rows with empty TA
A = A(rowsToKeep, :);

save wkspc_2 A S

%% Row Filtering
% List of variables to check
varsToCheck = {
    'POS_Ph1', 'POS_Ph2', 'POS_Ph3', 'POS_App', 'Duration_Ph1', 'Duration_Ph2',
'Duration_Ph3', 'Duration_App', ...
    'Cost_Ph1', 'Cost_Ph2', 'Cost_Ph3', 'Subjects_Ph1', 'Subjects_Ph2', 'Subjects_Ph3',
'Revenue_Peak', 'Revenue_Time'};

% Initialize logical index for rows to keep
rowsToKeep = false(height(A), 1);

for i = 1:height(A)
    disp(i)
    rowHasValue = false;
    for j = 1:numel(varsToCheck)
        val = A{i, varsToCheck{j}};
        if iscell(val)
            val = val{1};
        end
        if ~(isempty(val) || (ischar(val) && strcmp(val, "")) || (isnumeric(val) &&
isnan(val)))
            rowHasValue = true;
            break;
```

```
        end
    end
    rowsToKeep(i) = rowHasValue;
end
% Keep only rows with at least one non-empty value in the specified columns
A = A(rowsToKeep, :);

% Remove redundant rows
% List of columns to check for redundancy
colsToCheck = {
    'ID', 'Indication', 'POS_Ph1', 'POS_Ph2', 'POS_Ph3', 'POS_App', 'Duration_Ph1',
'Duration_Ph2', 'Duration_Ph3', 'Duration_App', ...
    'Cost_Ph1', 'Cost_Ph2', 'Cost_Ph3', 'Subjects_Ph1', 'Subjects_Ph2', 'Subjects_Ph3',
'Revenue_Peak', 'Revenue_Time'};

% Extract subtable with only the relevant columns
subA = A(:, colsToCheck);

% Standardize missing values
for j = 1:numel(colsToCheck)
    colName = colsToCheck{j};
    colData = subA.(colName);

    if isnumeric(colData)
        % Replace NaN with a placeholder
        colData(isnan(colData)) = -9999;
    elseif iscell(colData)
        % Replace empty strings or empty cells with a placeholder
        colData(cellfun(@(x) isempty(x) || (ischar(x) && strcmp(x, "")), colData)) =
{'missing'};
    elseif isstring(colData)
        % Replace empty strings with a placeholder
        colData(colData == "") = 'missing';
    end

    subA.(colName) = colData;
end

% Keep only unique rows
[~, uniqueIdx] = unique(subA, 'rows'); % Find unique rows based on standardized columns
A = A(uniqueIdx, :); % Keep only the unique rows in the full table

% Keep only specific status
A = A(ismember(A.Status, {'Approved', 'Filed', 'Marketed', 'Phase I', 'Phase II', 'Phase III',
'Phase 1', 'Phase 2', 'Phase 3'}), :);

save wkspc_3 A S

%% ----------------------------------------------------------------------------------------
----------------------------
% Now a similar exercise with the revenue dataset S

% Rename variables to simpler names
variableNames = {
    'Product', 'Product';
    'IndicationLevel1', 'TA';
    'IndicationLevel3', 'Indication';
    'WWPhase_Current_', 'WW_Phase';
    'USAPhase_Current_', 'US_Phase';
    'WWIndicationLaunch', 'WW_Launch';
    'USIndicationLaunch', 'US_Launch';};
```

```
% Define year columns
yearCols = {'x2004','x2005','x2006','x2007','x2008','x2009','x2010','x2011','x2012','x2013',
...
            'x2014','x2015','x2016','x2017','x2018','x2019','x2020','x2021','x2022','x2023',
...
            'x2024','x2025','x2026','x2027','x2028','x2029','x2030'};

% Redefine headers in S to preferred names
for i = 1:size(variableNames, 1)
    S.Properties.VariableNames{variableNames{i, 1}} = variableNames{i, 2};
end

% Remove columns in S that are not variableNames or years of revenue
selectNames = variableNames(:, 2); % Extract the original variable names from the first column
of variableNames
selectNames = [selectNames; S.Properties.VariableNames(yearCols)'];
varsToRemove = ~ismember(S.Properties.VariableNames, selectNames); % Identify which variables
in A are not in the originalNames list
S(:, varsToRemove) = [];% Remove those variables from S

% Remove rows that have no revenue across all years
R = table2array(S(:,yearCols));
R(isnan(R)) = 0;
R_row_sum = sum(R, 2);
S = S(R_row_sum > 0, :); % table with non-zero revenue rows

% Remove rows without launch date
S = S(~isnat(S.US_Launch), :);


% Initialize logical index for rows to keep
rowsToKeep = true(height(S), 1);

% Apply mapping and mark rows with empty TA for deletion
for i = 1:height(S)
    disp(i)
    ta = S.TA{i};
    if isempty(ta) || (isstring(ta) && ta == "")
        rowsToKeep(i) = false;  % Mark for deletion
    elseif isKey(taMap, ta)
        S.TA{i} = taMap(ta);    % Map to standardized category
    end
end

% Delete rows with empty TA
S = S(rowsToKeep, :);

% Calculate maximum revenue from available revenue curves
SS = table2array(S(:,yearCols));
S.Revenue_Peak = max(SS,[],2,'omitnan');

% Calculate the time to maximum revenue from the revenue curves, for those that have at least
a minimum, e.g., 10, number of entries
% Preallocate result array
numDrugs = size(SS, 1);
Revenue_Time = NaN(numDrugs, 1);

for i = 1:numDrugs
    rowData = SS(i, :);
```

```
    % Check for at least 10 non-NaN entries
    if sum(~isnan(rowData)) >= 10
        % Find index of first non-NaN value
        firstIdx = find(~isnan(rowData), 1, 'first');

        % Find index of max value (ignoring NaNs)
        [~, maxIdx] = max(rowData, [], 'omitnan');

        % Compute time to max relative to first valid year
        Revenue_Time(i) = maxIdx - firstIdx + 1;
    end
end

% Add to table S
S.Revenue_Time = Revenue_Time;

save wkspc_cleaned A S

writetable(S, 'Table_S.xlsx');
writetable(A, 'Table_A.xlsx');

end
```

```
function create_vars_table

%% NOTES
% Removed 'Hepatic & Biliary', 'Hormone', 'Miscellaneous', 'Sensory Organs' due to
insufficient data to inform statistical distribtions of variables

%% Initialize
load wkspc_cleaned A S

TA_categories = {'Blood', 'Cancer', 'Cardiovascular', 'Diabetes', 'Gastrointestinal',
'HIV',...
    'Immunology', 'Infections', 'Musculoskeletal', 'Neurology', 'Psychiatry',
'Reproduction',...
    'Respiratory', 'Skin', 'Surgery', 'Urinary Tract'};


var_names = [{'Variable'}, TA_categories]; % Correctly define the variable names

% Initialize the table with the appropriate size and variable types
numRows = 25;
numCols = length(var_names);
T = table('Size', [numRows, numCols], 'VariableTypes', repmat({'string'}, 1, numCols),
'VariableNames', var_names);

for i=1:length(TA_categories)
    disp(TA_categories{i});
    [p, ~] = create_TA_datasets(A, S, TA_categories{i}, 0, 1, 0);
    T = fillTable(T, p, TA_categories{i});
end

% Display the table
disp(T);

% Export the table to Excel
writetable(T, 'Table_T.xlsx');
```

```
end

%% Define a function to fill the table for each indication
function T = fillTable(T, p, therapeutic_area)

% Fill the table with the specified data
row = 1;

%% Number of drugs
T{row, 'Variable'} = {'Number of Drugs'};
T{row, therapeutic_area} = p.n;
row = row + 1;

%% POS Phase (Normal)
for i = 1:4
    pos_label = {'POS Phase 1', 'POS Phase 2', 'POS Phase 3', 'POS App'};
    T{row, 'Variable'} = pos_label(i);
    T{row, therapeutic_area} = round(p.pos.phase{i}.a,2,'significant');
    row = row + 1;
end

%% Duration Phase (Normal)
for i = 1:4
    pos_labels = {'Duration Phase 1', 'Duration Phase 2', 'Duration Phase 3', 'Duration App'};
    T{row, 'Variable'} = pos_labels(i);
    T{row, therapeutic_area} = round(p.duration.phase{i}.a,2,'significant');
    row = row + 1;
end

%% Cost Phase (Normal)
for i = 1:3
    T{row, 'Variable'} = {sprintf('Cost Phase %d', i)};
    T{row, therapeutic_area} = round(p.costs.phase{i}.a,2,'significant');
    row = row + 1;
end

%% Subject Phase (Normal)
for i = 1:3
    T{row, 'Variable'} = {sprintf('Subject Phase %d', i)};
    T{row, therapeutic_area} = round(p.subjects.phase{i}.a,2,'significant');
    row = row + 1;
end

%% Cost/Subject Phase (Normal)
for i = 1:3
    T{row, 'Variable'} = {sprintf('Cost/Subject Phase %d', i)};
    T{row, therapeutic_area} = round(p.cost_subject.phase{i}.a,2,'significant');
    row = row + 1;
end

%% Peak Revenue (Normal)
T{row, 'Variable'} = {'Peak Revenue'};
T{row, therapeutic_area} = round(p.revenue_peak.a,2,'significant');
row = row + 1;

%% Time to Peak Revenue (Normal)
T{row, 'Variable'} = {'Time to Peak Revenue'};
T{row, therapeutic_area} = round(p.revenue_time.a,2,'significant');
row = row + 1;

%% NPV Phases (Normal)
```

```
npv_labels = {'NPV PC', 'NPV Phase 1', 'NPV Phase 2', 'NPV Phase 3', 'NPV App'};
for i = 1:5
    T{row, 'Variable'} = npv_labels(i);
%      T{row, therapeutic_area} = {[num2str(round(p.npv{i}.a,2,'significant')),' / '
num2str(round(p.npv{i}.c,2,'significant'))]}; % mean / median
    T{row, therapeutic_area} = round(p.npv{i}.a,2,'significant'); % mean
    row = row + 1;
end

% % NPV App
% T{row, 'Variable'} = {'NPV App'};
% T{row, indication} = round(NPV_App,2,'significant');

end
```

```
function create_summary_figs

%% NOTES

%% Intialize data and layouts
close all;
% Load the data from the Excel file
data = readtable('Table_T.xlsx', 'Sheet', 'Sheet1','ReadRowNames', true,
'PreserveVariableNames', true);

% Convert all variables to double if possible
for i = 1:width(data)
    if iscell(data.(i))
        % Try converting cell to numeric
        try
            data.(i) = cellfun(@str2double, data.(i));
        catch
            warning('Column %s could not be converted to double.',
data.Properties.VariableNames{i});
        end
    end
end

% Extract the relevant columns for each plot
indications = data.Properties.VariableNames(1:end);

cost_phase_1 = data{strcmp(data.Variable, 'Cost Phase 1'), 1:end};
cost_phase_2 = data{strcmp(data.Variable, 'Cost Phase 2'), 1:end};
cost_phase_3 = data{strcmp(data.Variable, 'Cost Phase 3'), 1:end};

duration_phase_1 = data{strcmp(data.Variable, 'Duration Phase 1'), 1:end};
duration_phase_2 = data{strcmp(data.Variable, 'Duration Phase 2'), 1:end};
duration_phase_3 = data{strcmp(data.Variable, 'Duration Phase 3'), 1:end};

pos_phase_1 = data{strcmp(data.Variable, 'POS Phase 1'), 1:end};
pos_phase_2 = data{strcmp(data.Variable, 'POS Phase 2'), 1:end};
pos_phase_3 = data{strcmp(data.Variable, 'POS Phase 3'), 1:end};
pos_app = data{strcmp(data.Variable, 'POS App'), 1:end};

subjects_phase_1 = data{strcmp(data.Variable, 'Subject Phase 1'), 1:end};
subjects_phase_2 = data{strcmp(data.Variable, 'Subject Phase 2'), 1:end};
subjects_phase_3 = data{strcmp(data.Variable, 'Subject Phase 3'), 1:end};

cost_per_subject_phase_1 = data{strcmp(data.Variable, 'Cost/Subject Phase 1'), 1:end};
cost_per_subject_phase_2 = data{strcmp(data.Variable, 'Cost/Subject Phase 2'), 1:end};
```

```
cost_per_subject_phase_3 = data{strcmp(data.Variable, 'Cost/Subject Phase 3'), 1:end};

npv_pc = data{strcmp(data.Variable, 'NPV PC'), 1:end};
npv_phase_1 = data{strcmp(data.Variable, 'NPV Phase 1'), 1:end};
npv_phase_2 = data{strcmp(data.Variable, 'NPV Phase 2'), 1:end};
npv_phase_3 = data{strcmp(data.Variable, 'NPV Phase 3'), 1:end};
npv_app = data{strcmp(data.Variable, 'NPV App'), 1:end};

% Define a muted rainbow + brown/grey color palette
Colors = [
    0.80, 0.40, 0.40; % muted red
    0.90, 0.55, 0.35; % muted orange
    0.90, 0.75, 0.40; % muted yellow
    0.65, 0.80, 0.45; % muted yellow-green
    0.45, 0.75, 0.45; % muted green
    0.40, 0.75, 0.60; % muted turquoise
    0.40, 0.70, 0.75; % muted cyan
    0.45, 0.65, 0.80; % muted sky blue
    0.50, 0.55, 0.80; % muted blue
    0.60, 0.50, 0.80; % muted indigo
    0.70, 0.50, 0.75; % muted violet
    0.75, 0.50, 0.65; % muted magenta
    0.80, 0.50, 0.55; % muted rose
    0.60, 0.45, 0.35; % muted brown
    0.70, 0.60, 0.50; % muted tan
    0.75, 0.75, 0.75; % light grey
    0.50, 0.50, 0.50; % dark grey
    ];

% Create a tiled layout with reduced spacing
figure(1);
set(gcf, 'color', 'w');
pos = [274.6000 89 1.5336e+03 958.4000];
set(gcf, 'Position', pos);
tiledlayout(2, 3, 'TileSpacing', 'compact','Padding','compact');
axes_font_size = 9;
title_font_size = 10;

%% Cost
nexttile(3);
barh([cost_phase_1; cost_phase_2; cost_phase_3]', 'stacked', 'FaceAlpha', 0.5, 'EdgeColor',
'none');
set(gca, 'YTick', 1:length(indications), 'YTickLabel', indications, 'XScale', 'log',
'FontSize', axes_font_size);
xlabel('$ Million');
xlim([1 800]);
title('Cost', 'FontSize', title_font_size, 'FontWeight', 'bold');
for i = 1:length(indications)
    text(cost_phase_1(i)/2, i, num2str(cost_phase_1(i)), 'HorizontalAlignment', 'center',
'FontSize', axes_font_size-1);
    text(cost_phase_1(i) + cost_phase_2(i)/2, i, num2str(cost_phase_2(i)),
'HorizontalAlignment', 'center', 'FontSize', axes_font_size-1);
    text(cost_phase_1(i) + cost_phase_2(i) + cost_phase_3(i)/2, i, num2str(cost_phase_3(i)),
'HorizontalAlignment', 'center', 'FontSize', axes_font_size-1);
end
grid on
ax = gca;
ax.XGrid = 'on';
ax.YGrid = 'off';

%% Duration
```

```
nexttile(2);
interim = 6*ones(length(duration_phase_1),1)';
bbb = barh([duration_phase_1; interim; duration_phase_2; interim; duration_phase_3]',
'stacked', 'FaceAlpha', 0.5, 'EdgeColor', 'none');
ccc = lines;
bbb(1).FaceColor = ccc(1,:); bbb(1).FaceAlpha = 0.5;
bbb(2).FaceColor = [0.5 0.5 0.5];
bbb(3).FaceColor = ccc(2,:); bbb(1).FaceAlpha = 0.5;
bbb(4).FaceColor = [0.5 0.5 0.5];
bbb(5).FaceColor = ccc(3,:); bbb(1).FaceAlpha = 0.5;
set(gca, 'YTick', 1:length(indications), 'YTickLabel', indications, 'FontSize',
axes_font_size);
xlabel('Months');
title('Duration', 'FontSize', title_font_size, 'FontWeight', 'bold');
for i = 1:length(indications)
    text(duration_phase_1(i)/2, i, num2str(duration_phase_1(i)), 'HorizontalAlignment',
'center', 'FontSize', axes_font_size-1);
    text(duration_phase_1(i) + 6 + duration_phase_2(i)/2, i, num2str(duration_phase_2(i)),
'HorizontalAlignment', 'center', 'FontSize', axes_font_size-1);
    text(duration_phase_1(i) + 6 + duration_phase_2(i) + 6 + duration_phase_3(i)/2, i,
num2str(duration_phase_3(i)), 'HorizontalAlignment', 'center', 'FontSize', axes_font_size-1);
end
grid on
ax = gca;
ax.XGrid = 'on';
ax.YGrid = 'off';

%% Probability of Success
nexttile(1);
hold on;

% Define colors for each phase
phase_colors = lines;

% Plot each phase as a separate scatter series
for phase = 1:4
    switch phase
        case 1
            x = pos_phase_1;
        case 2
            x = pos_phase_2;
        case 3
            x = pos_phase_3;
        case 4
            x = pos_app;
    end
    y = 1:length(indications);
    scatter(x, y, 80, 'filled', 'MarkerFaceColor', phase_colors(phase,:), 'DisplayName',
['Phase ' num2str(phase)],'MarkerFaceAlpha',0.6);
end

set(gca, 'YTick', 1:length(indications), 'YTickLabel', indications, 'FontSize',
axes_font_size);
xlabel('Probability');
xlim([0 1.1]);
ylim([0.5 length(indications)+0.5]);
title('Probability of Success', 'FontSize', title_font_size, 'FontWeight', 'bold');
% hL = legend({'Phase 1', 'Phase 2', 'Phase 3', 'Approval'}, 'Location', 'eastoutside',
'FontSize', axes_font_size, 'Box', 'off');
% legendPos = get(hL, 'Position');
% legendPos(1) = legendPos(1) - 0.00;
```

```
% set(hL, 'Position', legendPos);
hold off;
grid on
ax = gca;
ax.XGrid = 'on';
ax.YGrid = 'on';
box on;

%% Peak Revenue
nexttile(6);
hold on;

% Extract relevant data
time_to_peak = data{'Time to Peak Revenue', :};
peak_revenue = data{'Peak Revenue', :};

% Scale marker sizes based on peak revenue
marker_sizes = 500 * sqrt(peak_revenue / max(peak_revenue));

% Create the bubble chart
for i = 1:length(indications)
    scatter(time_to_peak(i), peak_revenue(i), marker_sizes(i), 'MarkerFaceColor', Colors(i,:),
'MarkerEdgeColor', 'none', 'MarkerFaceAlpha', 0.7, 'DisplayName', indications{i});
end
hold off;

xlabel('Time to Peak (Years)');
ylabel('Peak ($ Million)');
title('Peak Revenue', 'FontSize', title_font_size, 'FontWeight', 'bold');
set(gca, 'FontSize', axes_font_size);
grid on;
box on;
xlim([0 12]);
ylim([0 1000]);
hL = legend('Location', 'eastoutside', 'FontSize', axes_font_size, 'Box', 'off');
legendPos = get(hL, 'Position');
legendPos(1) = legendPos(1) - 0.00;
set(hL, 'Position', legendPos);

%% Subjects
nexttile(4);
barh([subjects_phase_1; subjects_phase_2; subjects_phase_3]', 'stacked', 'FaceAlpha', 0.5,
'EdgeColor', 'none');
set(gca, 'YTick', 1:length(indications), 'YTickLabel', indications, 'XScale', 'log',
'FontSize', axes_font_size);
xlabel('Number');
title('Subjects', 'FontSize', title_font_size, 'FontWeight', 'bold');
for i = 1:length(indications)
    text(subjects_phase_1(i)/2, i, num2str(subjects_phase_1(i)), 'HorizontalAlignment',
'center', 'FontSize', axes_font_size-1);
    text(subjects_phase_1(i) + subjects_phase_2(i)/2, i, num2str(subjects_phase_2(i)),
'HorizontalAlignment', 'center', 'FontSize', axes_font_size-1);
    text(subjects_phase_1(i) + subjects_phase_2(i) + subjects_phase_3(i)/2, i,
num2str(subjects_phase_3(i)), 'HorizontalAlignment', 'center', 'FontSize', axes_font_size-1);
end
grid on;
ax = gca;
ax.XGrid = 'on';
ax.YGrid = 'off';
xlim([10 4000]);
```

```
%% Cost per Subject
nexttile(5);
hold on;

% Define colors for each phase
phase_colors = lines;

% Plot each phase as a separate scatter series
for phase = 1:3
    switch phase
        case 1
            x = cost_per_subject_phase_1;
        case 2
            x = cost_per_subject_phase_2;
        case 3
            x = cost_per_subject_phase_3;
    end
    y = 1:length(indications);
    scatter(x, y, 80, 'filled', 'MarkerFaceColor', phase_colors(phase,:), 'DisplayName',
['Phase ' num2str(phase)],'MarkerFaceAlpha',0.6);
end

set(gca, 'YTick', 1:length(indications), 'YTickLabel', indications, 'FontSize',
axes_font_size);
xlabel('$ Million');
xlim([0 0.8]);
ylim([0.5 length(indications)+0.5]);
title('Cost per Subject', 'FontSize', title_font_size, 'FontWeight', 'bold');
% hL = legend({'Phase 1', 'Phase 2', 'Phase 3'}, 'Location', 'eastoutside', 'FontSize',
axes_font_size, 'Box', 'off');
% legendPos = get(hL, 'Position');
% legendPos(1) = legendPos(1) - 0;
% set(hL, 'Position', legendPos);
hold off;
grid on
ax = gca;
ax.XGrid = 'on';
ax.YGrid = 'on';
box on;

%% Print the figure to pdf
set(gcf,'InvertHardCopy','off');
figSnapshotImage = ['Variables_Across_TAs_', date]; % Create the title string
set(gcf, 'Renderer', 'painters');
filename = [figSnapshotImage, '.pdf']; % Append the file extension
exportgraphics(gcf, filename, 'ContentType', 'vector'); % Export the figure

%% Net Present Value
figure(2);
set(gcf, 'color', 'w');
pos = [695.4000 349 767.2000 392.8000];
set(gcf, 'Position', pos);

hold on;
for i = 1:length(indications)
    plot(1:5, [npv_pc(i), npv_phase_1(i), npv_phase_2(i), npv_phase_3(i), npv_app(i)], '-o',
'Color', Colors(i,:), 'MarkerFaceColor', Colors(i,:), 'LineWidth', 1.5);
end
set(gca, 'XTick', 1:5, 'XTickLabel', {'Preclinical', 'Phase 1', 'Phase 2', 'Phase 3',
'Approval'}, 'FontSize', axes_font_size);
ylabel('NPV ($ Million)');
```

```
xlim([0.5 5.5]);
title('Net Present Value', 'FontSize',  title_font_size, 'FontWeight', 'bold');
lgd2 = legend(indications, 'Location', 'eastoutside', 'FontSize', axes_font_size);
set(lgd2, 'Box', 'off', 'Color', 'none');
hold off;
grid on
ax = gca;
ax.XGrid = 'on';
ax.YGrid = 'on';
box on;

%% Print the figure to pdf
set(gcf,'InvertHardCopy','off');
figSnapshotImage = ['NPV_Across_TAs_', date]; % Create the title string
set(gcf, 'Renderer', 'painters');
filename = [figSnapshotImage, '.pdf']; % Append the file extension
exportgraphics(gcf, filename, 'ContentType', 'vector'); % Export the figure

end
```

```
function T = create_sens_anlys_table

%% Initialize
load wkspc_cleaned A S

%%
therapeutic_area = 'Cancer';

tot_rev_multiplier = [1 1.05 1.1 1.2 1.4];
stdev_multiplier = [0 1 2];
multiplier = {'Base', '+ 5%', '+ 10%', '+ 20%', '+ 40%'};
var_names = [{'Variable'}, multiplier]; % Correctly define the variable names

% Initialize the table with the appropriate size and variable types
for i=1:length(stdev_multiplier)
    for j=1:length(tot_rev_multiplier)
        [p, SALES] = create_TA_datasets(A, S, therapeutic_area, stdev_multiplier(i),
tot_rev_multiplier(j), 1);
        for k=1:5
            T(j+(i-1)*length(tot_rev_multiplier),k) = round(p.npv{k}.a,2,'significant');
        end
        T(j+(i-1)*length(tot_rev_multiplier),6) = round(max(mean(SALES)),2,'significant');
        T(j+(i-1)*length(tot_rev_multiplier),7) = round(sum(mean(SALES)),2,'significant');
    end
end
T =  flipud(T);
close all
end
```

```
function [p, SALES] = create_TA_datasets(A, S, therapeutic_area, stdev_multiplier,
tot_rev_multiplier, SenAn)

%% NOTES
% function create_TA_datasets
% close all
% therapeutic_area = 'Cancer';
% stdev_multiplier = 0;
% tot_rev_multiplier = 1;
% SenAn = 0;
```

```
% load wkspc_cleaned A S

%% Initialize
% Define therapeutic area databases B and C
TA = therapeutic_area;
B = A(strcmp(TA, A.TA), :);
C = S(strcmp(TA, S.TA), :);

% Initialize
figure;
colors = lines;
phases = {'Ph1', 'Ph2', 'Ph3', 'App'};
set(gcf, 'color', 'w');
pos = [239.4,47.4,1646.4,1016.8];
set(gcf, 'Position', pos);
layout_cols = 5;
layout1 = tiledlayout(5, layout_cols, 'TileSpacing', 'Normal');
annotation('textbox', [0.1, 0.95, 0.8, 0.05], 'String', {[TA,' (n =
',num2str(size(B,1)),')']}, 'FontWeight', 'bold', 'HorizontalAlignment', 'center',
'VerticalAlignment', 'top', 'FontSize', 14, 'EdgeColor', 'none', 'FitBoxToText', 'on');
p.n = round(size(B,1));

%% Derive Variables
% Probability of Success - Updated!
col = 1;
x_label = 'probability';
y_label = 'counts';
X = [B.POS_Ph1, B.POS_Ph2, B.POS_Ph3, B.POS_App];
X(X == 1e-6) = NaN;
X(X == 1) = 0.99;
plot_boxplot(layout1, X, col, colors, phases, 'Probabilities of Success');
ylabel('probability');
for i = 1:size(X, 2)
    med = median(X(:,i),'omitnan');
    [a, b, c] = plot_distribution_new(layout1, layout_cols, X(:, i), i, 1, colors, phases,
x_label, y_label, med);
    p.pos.phase{i}.a = a;
    p.pos.phase{i}.b = b;
    p.pos.phase{i}.c = c;
    xlim([0 1.05]);
end

% Costs - Updated!
col = 2;
x_label = '$M';
y_label = 'counts';
X = [B.Cost_Ph1, B.Cost_Ph2, B.Cost_Ph3];
X(X == 1e-6) = NaN;
plot_boxplot(layout1, X, col, colors, phases, 'Costs');
ylabel('$M');
for i = 1:size(X, 2)
    Z = X(:,i);
    Z_filtered = Z(Z < nanmean(X(:, i)) + 3*nanstd(X(:, i)));
    med = median(X(:,i),'omitnan');
    [a, b, c] = plot_distribution_new(layout1, layout_cols, Z_filtered, i, col, colors,
phases, x_label, y_label, med);
    p.costs.phase{i}.a = a;
    p.costs.phase{i}.b = b;
    p.costs.phase{i}.c = c;
end
```

```
% Duration - Updated!
col = 3;
x_label = 'months';
y_label = 'counts';
X = [B.Duration_Ph1, B.Duration_Ph2, B.Duration_Ph3, B.Duration_App];
X(X == 1e-6) = NaN;
plot_boxplot(layout1, X, col, colors, phases, 'Duration');
ylabel('months');
for i = 1:size(X, 2)
    Z = X(:,i);
    Z_filtered = Z(Z < nanmean(X(:, i)) + 3*nanstd(X(:, i)));
    med = median(X(:,i),'omitnan');
    [a, b, c] = plot_distribution_new(layout1, layout_cols, Z_filtered, i, col, colors,
phases, x_label, y_label, med);
    p.duration.phase{i}.a = a;
    p.duration.phase{i}.b = b;
    p.duration.phase{i}.c = c;
end

% Subjects - Updated!
X = [B.Subjects_Ph1, B.Subjects_Ph2, B.Subjects_Ph3];
X(X == 1e-6) = NaN;
for i = 1:size(X, 2)
    [a, b, c] = generate_distribution_new(X(:, i));
    p.subjects.phase{i}.a = a;
    p.subjects.phase{i}.b = b;
    p.subjects.phase{i}.c = c;
end

% Cost/Subject - Updated!
B.Cost_Subject_Ph1 = B.Cost_Ph1./B.Subjects_Ph1;
B.Cost_Subject_Ph2 = B.Cost_Ph2./B.Subjects_Ph2;
B.Cost_Subject_Ph3 = B.Cost_Ph3./B.Subjects_Ph3;
X = [B.Cost_Subject_Ph1, B.Cost_Subject_Ph2, B.Cost_Subject_Ph3];
X(X == 1e-6) = NaN;
for i = 1:size(X, 2)
    Z = X(:,i);
    Z_filtered = Z(Z < nanmean(X(:, i)) + 3*nanstd(X(:, i)));
    [a, b, c] = generate_distribution_new(Z_filtered);
    p.cost_subject.phase{i}.a = a;
    p.cost_subject.phase{i}.b = b;
    p.cost_subject.phase{i}.c = c;
end

% Revenue - Updated!
col = 4;
X = [C.Revenue_Peak, C.Revenue_Time];
X(X == 1e-6) = NaN;
plot_boxplot(layout1, X, col, colors, {'Peak','Time to Peak'}, 'Revenue');
ylabel('$M');

for i = 1:size(X, 2)
    if i==1 % peak
        x_label = '$M';
        y_label = 'counts';
        Z = X(:,i);
        Z_filtered = Z(Z < nanmean(X(:, i)) + 3*nanstd(X(:, i)));
        med = median(X(:,i),'omitnan');
        [a, b, c] = plot_distribution_new(layout1, layout_cols, Z_filtered, i, col, colors,
{'Peak','Time'}, x_label, y_label, med);
        p.revenue_peak.a = a;
```

```
        p.revenue_peak.b = b;
        p.revenue_peak.c = c;

    else % time
        x_label = 'years';
        y_label = 'counts';
        med = median(X(:,i),'omitnan');
        [a, b, c] = plot_distribution_new(layout1, layout_cols, X(:, i), i, col, colors,
{'Peak','Time'}, x_label, y_label, med);
        p.revenue_time.a = a;
        p.revenue_time.b = b;
        p.revenue_time.c = c;
    end
end

%% Initialize Revenue
% Create a new table, T
T = S;
yearCols = {'x2004','x2005','x2006','x2007','x2008','x2009','x2010','x2011','x2012','x2013',
...
            'x2014','x2015','x2016','x2017','x2018','x2019','x2020','x2021','x2022','x2023',
...
            'x2024','x2025','x2026','x2027','x2028','x2029','x2030'};

% Remove rows without launch date
TR = table2array(T(:,yearCols)); % matrix of revenue values, including NaN

% Extract indication launch year
y = year(T.US_Launch);

% Remove old assets (before 2004)
old_assets_index = y < 2004;
T(old_assets_index, :) = [];
TR(old_assets_index, :) = [];
y(old_assets_index) = [];

% Shift revenue data to start from 2004
new_TR = NaN(size(TR, 1), 60);
for i = 1:length(y)
    shift = 2004 - y(i);
    old_row = TR(i, :);
    if shift > 0
        new_TR(i, shift+1:shift+length(old_row)) = old_row;
    else
        new_TR(i, 1:length(old_row) + shift) = old_row(-shift+1:end);
    end
end
TR = new_TR; % TR now contains extra columns to account for shifting of revenue curves

% Indication Level
% Get unique list of 'TA' and remove empty strings
index = find(contains(string(T.TA),TA));
I = T(index,:); % table
IR = TR(index,:); % matrix

% Remove columns that are all NaN
IR(:, all(isnan(IR))) = [];

% Generate mean-normalized version
IR_norm = IR; % initalize matrix
for i=1:size(IR,1)
```

```
    IR_norm(i,:) = IR(i,:)./(nanmean(IR(i,:)));
end

nexttile(layout1, 19);
for i=1:size(IR_norm,1)
    plot(IR_norm(i,:)); hold on;
end
xlabel('years'); ylabel('$ normalized'); title(['Time-Adjusted, Normalized (n =
',num2str(size(IR,1)),')']);
xlim([0 30]);
ylim([0 inf]);

%% Model Revenue Curve as Piecewise Function
data = IR_norm;

% Initialize arrays to store the time and data values
time_values = [];
data_values = [];

% Loop through each column and extract non-NaN values
for i = 1:size(IR_norm, 2)
    col = IR_norm(:, i);
    non_nan_idx = ~isnan(col);
    time_values = [time_values; i * ones(sum(non_nan_idx), 1)];
    data_values = [data_values; col(non_nan_idx)];
end

% Parameters
time_ramp = p.revenue_time.a; % years from 0 to peak; default = 4
peak = mean(data_values);
time_at_peak = 8; % years at peak
time_end = 30; % end time
initial_guess = [peak, time_ramp, time_ramp + time_at_peak, time_end]; % Initial guess for b

% Define the piecewise continuous function
% b(1) = plateau height, i.e., peak
% b(2) = time to peak (end of ramp)
% b(3) = time at end of plateau
% b(4) = time at end of decline
piecewise_function = @(b, x) ...
    (x < b(2)) .* (x * (b(1) / b(2))) + ...
    (x >= b(2) & x < b(3)) .* b(1) + ...
    (x >= b(3) & x <= b(4)) .* (b(1) - (x - b(3)) * (b(1) / (b(4) - b(3))));

% Assign the data
x = time_values;
y = data_values;

% Fit the model (based on normalized and time shifted data)
opts = statset('nlinfit');
opts.RobustWgtFun = 'bisquare';
b0 = initial_guess;
[b, R, ~, CovB, MSE] = nlinfit(x, y, piecewise_function, b0, opts);

% Important Assumption: If the data is so sparse that a curve can't be generated, force
default values
if b(4) > 30
    b(4) = 30; % total duration
end
if b(2) < 2
    b(2) = 5; % time to peak
```

```
end
if b(3) - b(2) > 20 || b(3) - b(2) < 2
    b(3) = b(2) + 5; % duration of peak
end

% Generate predictions and confidence intervals
xrange = min(x):0.01:max(x);
[ypred, delta] = nlpredci(piecewise_function, xrange, b, R, 'Covar', CovB, 'MSE', MSE,
'SimOpt', 'on');
lower = ypred - delta;
upper = ypred + delta;

% Plot the data and the fitted function with confidence intervals
nexttile(layout1, 24);
plot(x, y, '.', 'Color', [0.678, 0.847, 0.902], 'DisplayName', 'Observed data'); % observed
data
hold on;
plot(xrange, ypred, 'Color', colors(1,:), 'LineWidth', 2, 'DisplayName', 'Fitted function');
fill([xrange, fliplr(xrange)], [lower, fliplr(upper)], 'r', 'FaceAlpha', 0.2, 'EdgeColor',
'none', 'DisplayName', '95% Confidence Interval');
xlabel('years');
ylabel('$ normalized');
hold off;
xlim([0 30]);
ylim([0 inf]);

% Add text with the values of b(1), b(2), b(2)+b(3), and b(4)-b(3)
ramp_time = round(b(2), 2, 'significant');
peak_time = round(b(3) - b(2), 2, 'significant');
decline_time = round(b(4) - b(3), 2, 'significant');
annotation_text = {['Ramp Time = ', num2str(ramp_time),' years']...
    ['Peak Duration = ', num2str(peak_time),' years'],['Decline Time = ',
num2str(decline_time),' years']};
text('Units', 'normalized', 'Position', [0.05, 0.75], 'String', annotation_text, 'FontSize',
8, 'BackgroundColor', 'none');
title(['Piecewise Model Fit (n = ',num2str(size(IR,1)),')']);

%% NPV ---> UPDATED TO INCLUDE DISTRIBUTION ON NPV_APP
for i=1:1000
    t = 1:round(max(x));
    [~, idx] = ismember(t, xrange);

    mean_peak = p.revenue_peak.a;
    stdev_peak = p.revenue_peak.b;
    median_peak = p.revenue_peak.c;
    lb_peak = 0.01*mean_peak;
    ub_peak = p.revenue_peak.a + 2*p.revenue_peak.b;

    if SenAn > 0 % this is for when "create_sens_anlys_table.m" is executed
        example_peak = mean_peak;
    else
        example_peak = generate_within_range_new(mean_peak, stdev_peak, lb_peak, ub_peak);
    end
    example_peak = (example_peak + stdev_multiplier*stdev_peak)*tot_rev_multiplier;

    sales = ypred(idx)*example_peak/b(1);
    SALES(i,:) = sales;
    t = [0 t];
    sales = [0 sales];
    discount_rate = 0.15;
    cogs = 0.25;
```

```
    sga = 0.20;
    NPV = 0;
    for j=1:length(t)
        net_cash(j) = sales(j) - sales(j).*(cogs + sga);
        NPV(j) = NPV(end) + net_cash(j)/(( 1 + discount_rate).^t(j));
    end
    NPV_model_fit = NPV(end); % NPV(t) = Sum [ Net Cash / (1 + Discount Rate)^t ]

    POS_Ph1 = generate_within_range_new(p.pos.phase{1}.a, p.pos.phase{1}.b, 0, 1);
    POS_Ph2 = generate_within_range_new(p.pos.phase{2}.a, p.pos.phase{2}.b, 0, 1);
    POS_Ph3 = generate_within_range_new(p.pos.phase{3}.a, p.pos.phase{3}.b, 0, 1);
    POS_App = generate_within_range_new(p.pos.phase{4}.a, p.pos.phase{4}.b, 0, 1);

    Cost_Ph1 = generate_within_range_new(p.costs.phase{1}.a, p.costs.phase{1}.b, 0, 250);
    Cost_Ph2 = generate_within_range_new(p.costs.phase{2}.a, p.costs.phase{2}.b, 0, 500);
    Cost_Ph3 = generate_within_range_new(p.costs.phase{3}.a, p.costs.phase{3}.b, 0, 1000);
    Cost_App = 2.5;

    Duration_Ph1 = generate_within_range_new(p.duration.phase{1}.a, p.pos.phase{1}.b, 0,
120)./12;
    Duration_Ph2 = generate_within_range_new(p.duration.phase{2}.a, p.pos.phase{2}.b, 0,
120)./12;
    Duration_Ph3 = generate_within_range_new(p.duration.phase{3}.a, p.pos.phase{3}.b, 0,
120)./12;
    Duration_App = generate_within_range_new(p.duration.phase{4}.a, p.pos.phase{4}.b, 0,
120)./12;

    Discount_Rate = discount_rate;

    Discounted_Cost_Ph1 = Cost_Ph1./((1 + Discount_Rate).^(Duration_Ph1 + 0.5));
    Discounted_Cost_Ph2 = Cost_Ph2./((1 + Discount_Rate).^(Duration_Ph2 + 0.5)) +
Discounted_Cost_Ph1;
    Discounted_Cost_Ph3 = Cost_Ph3./((1 + Discount_Rate).^(Duration_Ph3 + 0.5)) +
Discounted_Cost_Ph2;
    Discounted_Cost_App = Cost_App./((1 + Discount_Rate).^Duration_App) + Discounted_Cost_Ph3;

    NPV_App(i) = NPV_model_fit;
    NPV_Ph3(i) = NPV_App(i)*POS_App - Discounted_Cost_App*(1 - POS_App);
    NPV_Ph2(i) = NPV_Ph3(i)*POS_Ph3 - Discounted_Cost_Ph3*(1 - POS_Ph3);
    NPV_Ph1(i) = NPV_Ph2(i)*POS_Ph2 - Discounted_Cost_Ph2*(1 - POS_Ph2);
    NPV_PC(i) = NPV_Ph1(i)*POS_Ph1 - Discounted_Cost_Ph1*(1 - POS_Ph1);
end

%% NPV ---> UPDATED TO INCLUDE DISTRIBUTION ON NPV_APP
col = 5;
x_label = '$M';
y_label = 'counts';
npv_phases = {'PC', 'Ph1', 'Ph2', 'Ph3', 'App'};
X = [NPV_PC', NPV_Ph1', NPV_Ph2', NPV_Ph3', NPV_App'];
X(X == 1e-6) = NaN;

for i = 1:size(X, 2)
    data = X(:, i);
    data_clean = data(~isnan(data));
    pd = fitdist(data_clean, 'Normal');
    a = round(pd.mu, 2);
    b = round(pd.sigma, 2);
    p.npv{i}.a = a;
    p.npv{i}.b = b;
end
```

```
format long
L =  lines;
npv_colors = [L(5,:); L(1:4,:)];
plot_boxplot(layout1, X, col, npv_colors, npv_phases, ['NPV']);
ylabel('$M');
npv_phases = {'Ph1', 'Ph2', 'Ph3', 'App'};
X = [NPV_Ph1', NPV_Ph2', NPV_Ph3', NPV_App'];
X(X == 1e-6) = NaN;
for i = 1:size(X, 2)
    med = median(X(:,i),'omitnan');
    [~, ~, ~] = plot_distribution_new(layout1, layout_cols, X(:, i), i, col, colors,
npv_phases, x_label, y_label, med);
    xlim([min(min(X)) max(max(X))]);
end

%% Print the figure to pdf
set(gcf,'InvertHardCopy','off');
figSnapshotImage = [char(TA), '_', date]; % Create the title string
set(gcf, 'Renderer', 'painters');
filename = [figSnapshotImage, '.pdf']; % Append the file extension
exportgraphics(gcf, filename, 'ContentType', 'vector'); % Export the figure

end

%% Function to Generate a Given Decision Tree Variable from a Distribution - Updated!
function value = generate_within_range_new(a, b, lb, ub)
while true
    value = normrnd(a, b);
    if value >= lb && value <= ub
        break;
    end
end
end

%% Function to plot box plot - Updated!
function plot_boxplot(layout, data, col, colors, phases, var)
    nexttile(layout, col)
    hold on;
    max_whisker = -inf;
    for i = 1:size(data, 2)
        boxchart(ones(size(data(:, i))) * i, data(:, i), 'BoxFaceColor', colors(i, :),
'MarkerColor', colors(i, :));

        % Calculate and display the median value
        median_value = median(data(~isnan(data(:, i)), i));
        text(i, median_value, num2str(round(median_value, 2, 'significant')),
'HorizontalAlignment', 'center', 'VerticalAlignment', 'bottom', 'Color', colors(i, :),
'FontSize', 8);

        % Calculate whisker values
        q1 = quantile(data(~isnan(data(:, i)), i), 0.25);
        q3 = quantile(data(~isnan(data(:, i)), i), 0.75);
        iqr = q3 - q1;
        upper_whisker = min(max(data(~isnan(data(:, i)), i)), q3 + 1.5 * iqr);

        % Update max_whisker to the highest whisker value
        max_whisker = max(max_whisker, upper_whisker);

        % Calculate the number of data points
        num_points(i) = sum(~isnan(data(:, i)));
    end
```

```
    xticks(1:size(data, 2));
    xticklabels(phases(1:size(data, 2)));
    title(var, 'Units', 'normalized', 'Position', [0.5, 1.1, 0], 'fontsize', 12);
    hold off;
    ylim([min(ylim), max_whisker * 1.1]);

    % Add text string of number of points under each x-axis tick label
    for i = 1:size(data, 2)
        text(i, min(ylim) - (max(ylim) - min(ylim))*0.25, ['(', num2str(num_points(i)), ')'],
'HorizontalAlignment', 'center', 'VerticalAlignment', 'middle','fontsize', 8);
    end
end

%% Function to plot distribution - Updated!
function [a, b, c] = plot_distribution_new(layout, layout_cols, data, i, col, colors, phases,
x_label, y_label, med)

    % Scale PDF to match histogram
    scale_pdf_to_histogram = @(pdf_values, hist_values, bin_width) pdf_values *
sum(hist_values) * bin_width;

    % Prepare the plot
    nexttile(layout, (i) * layout_cols + col);
    data_clean = data(~isnan(data));

    % Fit the distribution

    pd = fitdist(data_clean, 'Normal');
    a = round(pd.mu, 2);
    b = round(pd.sigma, 2);
    c = med;

    % Plot the histogram and PDF
    x = linspace(min(data_clean), max(data_clean), 100);
    y = pdf(pd, x);
    hist = histogram(data_clean, 15, 'FaceColor', colors(i, :));
    bin_width = hist.BinWidth;
    hist_values = hist.Values;
    y_scaled = scale_pdf_to_histogram(y, hist_values, bin_width);
    hold on;
    plot(x, y_scaled, 'Color', colors(i, :), 'LineWidth', 2);
    hold off;
    xlabel(x_label);
    ylabel(y_label);
    ylim([0 max([max(hist_values); max(y_scaled)])*1.1]);

    % Add title and limits
    title([phases{i}, ': \mu = ', num2str(round(a, 2, 'significant')), ', \sigma = ',
num2str(round(b, 2, 'significant')), '; M = ', num2str(round(c, 2, 'significant'))]);
end

%% Function to generate distribution
function [a, b, c] = generate_distribution_new(data)

data_clean = data(~isnan(data));

% Fit the distribution
pd = fitdist(data_clean, 'Normal');
a = round(pd.mu, 2);
b = round(pd.sigma, 2);
c = round(median(data_clean),2);
```

```
end
```