Пловдивски университет „Паисий Хилендарски“

Факултет по Математика и информатика
Софтуерно инженерство, IV курс, редовно обучение

Верификация и валидиране на софтуер

КУРСОВА РАБОТА

върху

моделите за моделиране на надеждност на

софтуер

Gompertz и Cheng-Pham

Изготвили:

Олег Заим – 1701321109
Степан Серт – 1701321107

Ще разгледами две изградени оценки на софтуерна от литературата "Handbook of software reliability engineering" глава 7 "Software Reliability Measurement Experience" Allen P. Nikora and Michael R. Lyu.
J1, J4
1. J1(Наричан по-долу Софтуер 1)

1	**6**	**6**
2	**1**	**7**
3	**1**	**8**
4	**0**	**8**
5	**1**	**9**
6	**3**	**12**
7	**0**	**12**
8	**5**	**17**
9	**6**	**23**
10	**1**	**24**
11	**0**	**24**
12	**3**	**27**
13	**9**	**36**
14	**3**	**39**
15	**2**	**41**
16	**3**	**44**
17	**0**	**44**
18	**2**	**46**
19	**4**	**50**
20	**0**	**50**
21	**0**	**50**
22	**0**	**50**
23	**0**	**50**
24	**0**	**50**
25	**2**	**52**
26	**0**	**52**
27	**0**	**52**
28	**2**	**54**
29	**3**	**57**
30	**11**	**68**
31	**5**	**73**
32	**3**	**76**
33	**2**	**78**
34	**4**	**82**
35	**2**	**84**
36	**0**	**84**
37	**1**	**85**

38	**2**	**87**
39	**0**	**87**
40	**1**	**88**
41	**3**	**91**
42	**0**	**91**
43	**1**	**92**
44	**6**	**98**
45	**2**	**100**
46	**0**	**100**
47	**5**	**105**
48	**10**	**115**
49	**3**	**118**
50	**3**	**121**
51	**2**	**123**
52	**1**	**124**
53	**0**	**124**
54	**3**	**127**
55	**0**	**127**
56	**2**	**129**
57	**0**	**129**
58	**1**	**130**
59	**0**	**130**
60	**1**	**131**
61	**0**	**131**
62	**2**	**133**

J4(Наричан по-долу Софтуер 2)

1	**0**	**0**
2	**0**	**0**
3	**1**	**1**
4	**0**	**1**
5	**0**	**1**
6	**0**	**1**
7	**0**	**1**
8	**0**	**1**
9	**0**	**1**
10	**1**	**2**
11	**0**	**2**
12	**0**	**2**
13	**1**	**3**
14	**2**	**5**
15	**2**	**7**
16	**0**	**7**

17	**1**	**8**
18	**2**	**10**
19	**0**	**10**
20	**0**	**10**
21	**2**	**12**
22	**1**	**13**
23	**3**	**16**
24	**0**	**16**
25	**0**	**16**
26	**5**	**21**
27	**2**	**23**
28	**0**	**23**
29	**0**	**23**
30	**1**	**24**
31	**3**	**27**
32	**2**	**29**
33	**0**	**29**
34	**2**	**31**
35	**0**	**31**
36	**0**	**31**
37	**2**	**33**
38	**1**	**34**
39	**0**	**34**
40	**0**	**34**
41	**0**	**34**
42	**0**	**34**
43	**0**	**34**
44	**0**	**34**
45	**0**	**34**
46	**0**	**34**
47	**0**	**34**
48	**1**	**35**
49	**2**	**37**
50	**0**	**37**
51	**0**	**37**
52	**5**	**42**
53	**2**	**44**
54	**0**	**44**
55	**1**	**45**
56	**6**	**51**
57	**8**	**59**
58	**2**	**61**
59	**1**	**62**
60	**6**	**68**
61	**8**	**76**

62	**8**	**84**
63	**6**	**90**
64	**3**	**93**
65	**4**	**97**
66	**3**	**100**
67	**1**	**101**
68	**1**	**102**
69	**1**	**103**
70	**1**	**104**
71	**5**	**109**
72	**2**	**111**
73	**1**	**112**
74	**1**	**113**
75	**1**	**114**
76	**2**	**116**
77	**0**	**116**
78	**1**	**117**
79	**1**	**118**
80	**1**	**119**
81	**1**	**120**
82	**0**	**120**
83	**1**	**121**
84	**1**	**122**
85	**1**	**123**
86	**2**	**125**
87	**2**	**127**
88	**3**	**130**
89	**3**	**133**
90	**5**	**138**
91	**1**	**139**
92	**2**	**141**
93	**3**	**144**
94	**1**	**145**
95	**1**	**146**
96	**1**	**147**
97	**2**	**149**
98	**0**	**149**
99	**1**	**150**
100	**4**	**154**
101	**3**	**157**
102	**0**	**157**
103	**1**	**158**
104	**8**	**166**
105	**4**	**170**
106	**7**	**177**

107	**1**	**178**
108	**1**	**179**
109	**2**	**181**
110	**1**	**182**
111	**0**	**182**
112	**1**	**183**

Оценките са направени на база открити грешки в софтуера за определен период от време.

И в двете оценки се отбелязват дни без открити грешки. В отделна колона са въведени общият брой грешки до момента.

При Софтуер 1 са записани данни за 62 дни, а при Софтер 2 – за 112 дни.

От таблиците се забелязва, че общия брой грешки в Софтуер 2 е много по-голям, от колкото в Софтуер 1 – съответно 183 към 133, пресметнато за равен период (62 дни) – 133 към 84, което е 1.58 пъти по-малко открити дефекти.

Сравнявайки денят с най-много грешки от Софтуер 1 с този от Софтуер, съответно ден 30 (11 грешки) и ден 57(8 грешки), Софтуер 1 води по брой грешки за един ден.

При Софтуер 1 се забелязва, че много грешки се откриват още в първите дни на тестването, докато при Софтуер 2 в началото са малко на брой или изобщо няма.

Gompertz

$$M(t) = \omega a^{b^t}$$

- M(t): Комулативният брой софтуерни грешки, открити за време t >=0
- w: горната граница, на която надеждността се приближава асимптотично като T ∞, или максималната надеждност, която може да бъде постигната (~ w^*a)
- b: индикатора за модела на растеж
- t: индикация за време, старт или номер на етап, t>0

$$M(t) = \omega e^{-\alpha e^{-\beta t}}$$

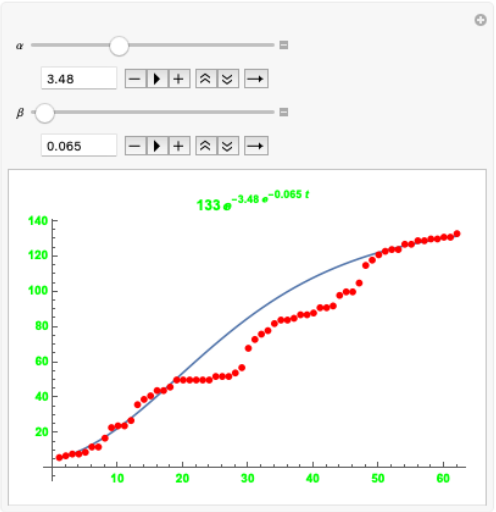
- α: параметър при поява на грешка
- β: коефициент на надеждност на системата

Софтуер 1

THE GOMPERTZ SOFTWARE RELIABILITY MODEL

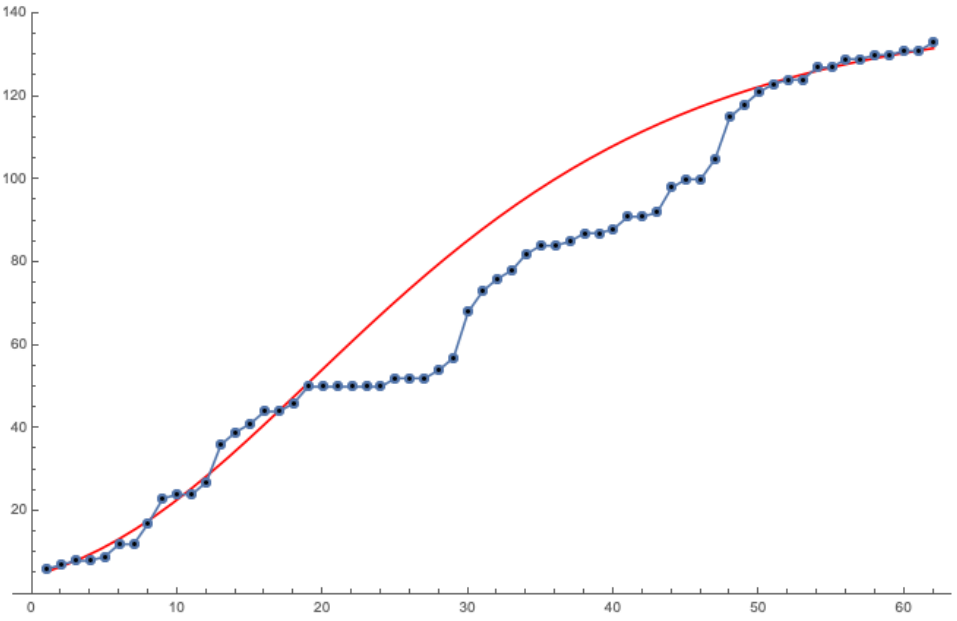
```
In[1217]:= j1 = Import["/Users/olegzaim/Downloads/Верефикация и Валидиране на Софтуер/FMI-SE-4-2-Software_Reliability-master/J1_data.csv"];
Manipulate[
 Dyn ow[Plot[f[t], {t, 1, 62}, (* функция на времето на изпитване *)
    LabelStyle → Directive[Green, Bold], (* форматиране на динамичната графика*)
    PlotLabel → 133 * Exp[-α * Exp[-β * t]]],
   ListPlot[ j1,
    PlotStyle → Red,
    PlotMarkers → {Automatic, Small}],
   PlotRange → {Automatic, {0, 133}},
   AxesOrigin → {0, 0}],
 {{α, 0}, 0, 10, Appearance → "Open" },
 {{β, 0}, 0.01, 10, Appearance → "Open" },
 Initialization :→ (f[t_] := 140 * Exp[-α * Exp[-β * t]])
]
```

Out[1218]=



In[1230]:=
```
α = 3.48;
β = 0.065;
f[t_] := 140 * Exp[-α * Exp[-β * t]];
linechart = Plot[f[t], {t, 1, 62}, Epilog → Map[Point, j1], PlotStyle → {Red}, PlotRange → {0, 140}];
Show[linechart, ListPlot[j1, Joined → True, Mesh → Full, MeshStyle → Directive[PointSize[Large], Thick]]]
```

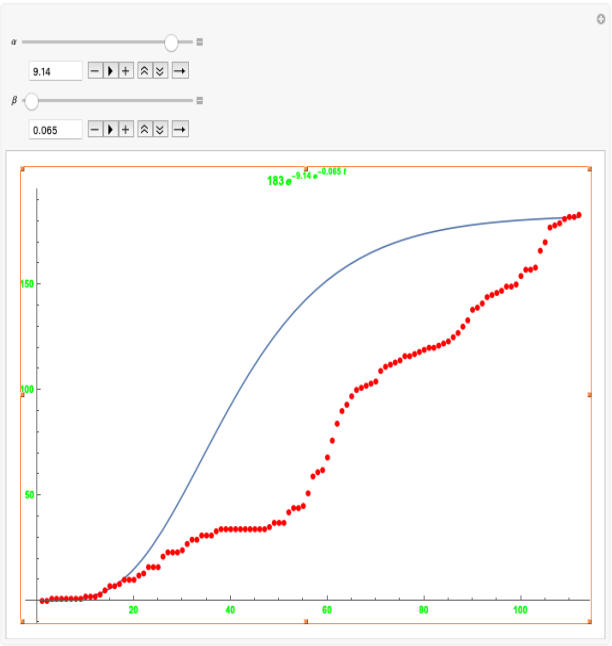
Out[1234]=



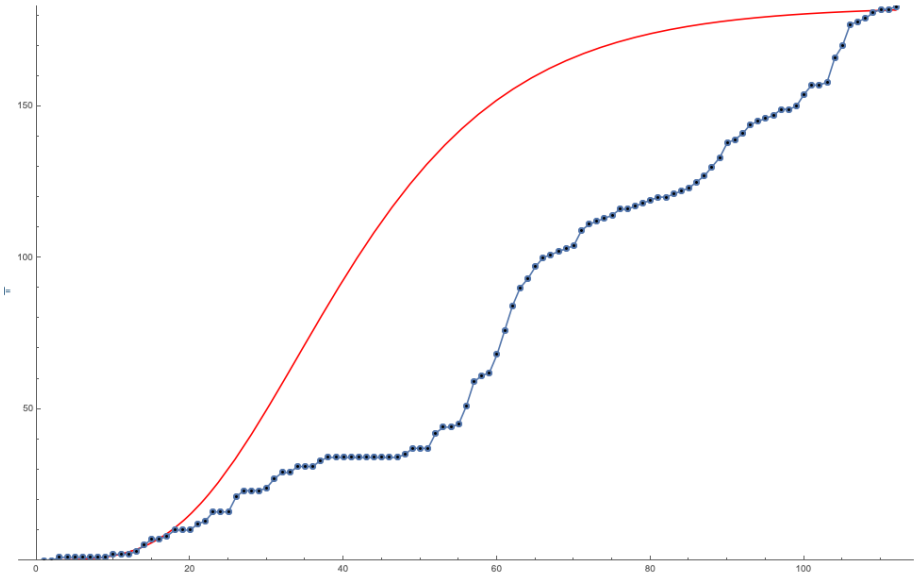
Софтуер 2

```
j4 = Import["/Users/olegzaim/Downloads/Верефикация и Валидиране на Софтуер/FMI-SE-4-2-Software_Reliability-master/J4_data.csv"];
Manipulate[
 Dynamic@Show[Plot[f[t], {t, 1, 112}, (* функция на времето на изпитване *)
    LabelStyle → Directive[Green, Bold], (* форматиране на динамичната графика*)
    PlotLabel → 183 * Exp[-α * Exp[-β * t]]],
   ListPlot[ j4,
    PlotStyle → Red,
    PlotMarkers → {Automatic, Small}],
   PlotRange → {Automatic, {0, 185}},
   AxesOrigin → {0, 0}],
 {{α, 0}, 0, 10, Appearance → "Open"},
 {{β, 0}, 0.01, 10, Appearance → "Open"},
 Initialization :→ (f[t_] := 183 * Exp[-α * Exp[-β * t]])
]
```

Out[3511]=



```
α = 9.14;
β = 0.065;
f[t_] := 183 * Exp[-α * Exp[-β * t]];
j4 = Import["/Users/olegzaim/Downloads/Верификация и Валидиране на Софтуер/FMI-SE-4-2-Software_Reliability-master/J4_data.csv"];
linechart = Plot[f[t], {t, 1, 112}, Epilog → Map[Point, j4], PlotStyle → {Red}, PlotRange → {0, 183}];
Show[linechart, ListPlot[j4, Joined → True, Mesh → Full, MeshStyle → Directive[PointSize[Large], Thick]]]
```



Cheng-Pham

$$M(t) = a(1 - \left(\frac{\beta}{\beta + \ln\frac{a + e^{bt}}{a+1}}\right)^{\alpha}$$

С използване на функцията FindFit получаваме числените стойности на параметрите b, α и β за списъка с входни данни при зададен израз – функцията на моела на Cheng-Pham.

```
model =  α * (1- β/ β + Log[(a+ Exp[b+t])/(a+1)])) ^  α
fit = FindFit[data,  model, [b, α,  β], t]
```

Софтуер 1

```
modelmodel = a * (1 - β / (β + Log[(a + Exp[b * t]) / (a + 1)])) ^ α
a = 140
fitfit = FindFit[j1, modelmodel, {b, α, β}, t]
β = 0.3535
α = 49.346
a = 140
b = 1.203
H[t_] := a * (1 - β / (β + Log[(a + Exp[b * t]) / (a + 1)])) ^ α
H1[t_] := b / (1 + a * Exp[-b * t])
H2[t_] := a
g1 = Plot[H[t], {t, 0, 62}, PlotRange → {0, 140}, AspectRatio → 0.6, PlotStyle → {Red}];
g2 = Plot[H1[t], {t, 0, 62}, PlotRange → {0, 140}, AspectRatio → 0.6, PlotStyle → {Blue}];
g3 = Plot[H2[t], {t, 0.1, 62}, PlotRange → {0, 140}, AspectRatio → 0.6, PlotStyle → {Dashed}];
Show[g1, g2, g3, ListPlot[j1, Joined → True, Mesh → Full, MeshStyle → Directive[PointSize[Large], Thick]]]
```

Out[1960]= $\left\{a \left(1 - \frac{\beta}{\beta + \text{Log}\left[\frac{a + e^{bt}}{1 + a}\right]}\right)^{\alpha}\right\}$

Out[1961]= 140

FindFit: Failed to converge to the requested accuracy or precision within 100 iterations.

Out[1962]= {b → 1.20336, α → 49.3467, β → 0.353559}

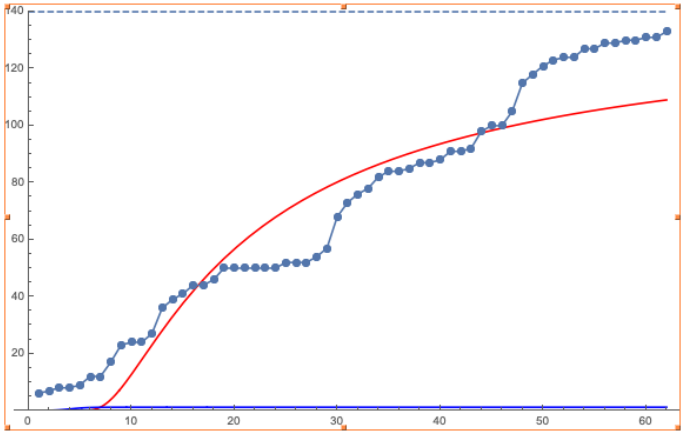
Out[1963]= 0.3535

Out[1964]= 49.346

Out[1965]= 140

Out[1966]= 1.203

Out[1973]=



Софтуер 2

In[3490]:=
```
j4 = Import["/Users/olegzaim/Downloads/Верефикация и Валидиране на Софтуер/FMI-SE-4-2-Software_Reliability-master/J4_data.csv"];
Clear[β]
Clear[b]
Clear[a]
Clear[α]
modelmodel = a * (1 - β / (β + Log[(a + Exp[b * t]) / (a + 1)])) ^ α
a = 198
fitfit = FindFit[j4, modelmodel, {b, α, β}, t]
β = 0.18
α = 1.028
a = 198
b = 0.053
H[t_] := a * (1 - β / (β + Log[(a + Exp[b * t]) / (a + 1)])) ^ α
H1[t_] := b / (1 + a * Exp[-b * t])
H2[t_] := a
g1 = Plot[H[t], {t, 0, 114}, PlotRange → {0, 190}, AspectRatio → 0.6, PlotStyle → {Red}];
g2 = Plot[H1[t], {t, 0, 114}, PlotRange → {0, 190}, AspectRatio → 0.6, PlotStyle → {Blue}];
g3 = Plot[H2[t], {t, 0.1, 114}, PlotRange → {0, 190}, AspectRatio → 0.6, PlotStyle → {Dashed}];
Show[g1, g2, g3, ListPlot[j4, Joined → True, Mesh → Full, MeshStyle → Directive[PointSize[Large], Thick]]]
```

Out[3495]=
$$a \left(1 - \frac{\beta}{\beta + \text{Log}\left[\frac{a + e^{bt}}{1 + a}\right]}\right)^{\alpha}$$

Out[3496]= 198

Out[3497]= {b → 0.0534186, α → 1.02897, β → 0.185076}

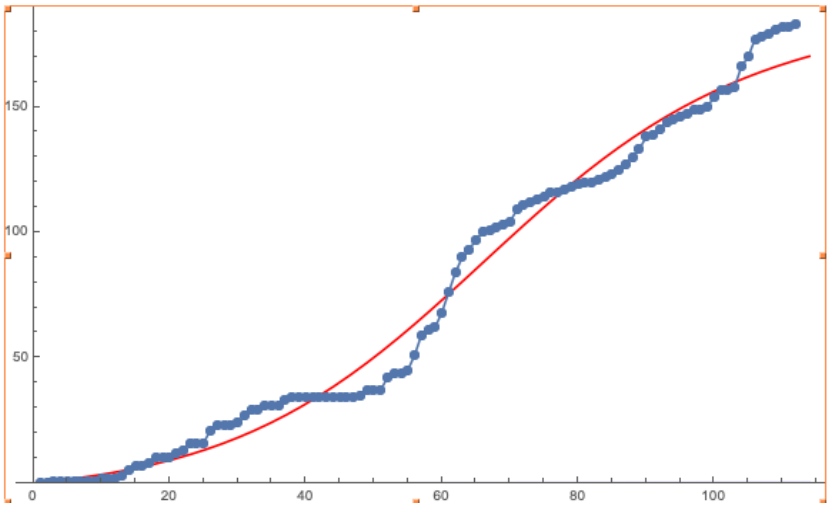
Out[3498]= 0.18

Out[3499]= 1.028

Out[3500]= 198

Out[3501]= 0.053

Out[3508]=



Извод:

Според първата графика Софтуер 1 е надежден. Почвайки от първата и до 19 седмица то се доближава до модела на Gompertz. От 19 и до 29 седмица виждаме че се откриват малко на брой грешки. Обаче след това почвайки от 30 седмица и до 50 виждаме че имаме скок на броя грешките. От 50 седмца и до края, броя на грешките онтово се изровниха с модела на Gompertz.

Анализираки втория график виждаме че първите 18 седмици съвпадат с модела на Gompertz, после се вижда че от 20 седмица и до 38 се увеличава броя на грешки и се разминава с графика. Следващите 16 седмици виждаме че софтуера е много надежден тъй като има по 1-2 грешки или изобщо няма. Обаче от тук и до 66 седмица виждаме силно увелечение на броя грешките, софтуера стана крайно не стабилен. След това до 86 седмица драстично се намалиха грешките и софтуера стано относително стабилен. От 86 седмица и до 107 наблюдаваме пак силно увелечаване на броя грешките. До края на изучаваните седмици софтуера се стабелизира и максимално се доблежи до функцията на Gompertz. Можем да кажем че софтуера е надежден въз основа на това че поне три последни седмци се доблежиха до модела на Gompertz.

Можем да кажем че Софтуер 2 е по-надежден от Сфотуер 1, защото за Софтуер 1 изкуствено увелечихме броя на грешките за да съвпада с графика на Gompertz.

И за двата софтуера този Gompertz модела работи добре. Той дава възможност да се предскаже какво би се случило в следващ етап от развитието на системата.

Модела Gompertz работи по-добре от Cheng-Pham конкретно за тези два софтуера.