kappa(2)분포를 이용한 소프트웨어 성장모형에 관한 연구

김희철*, 이상식**
*남서울대학교 산업정보시스템공학부
**송호대학 의료정보과
e-mail:*kim1458@nsu.ac.kr, **leess@songho.ac.kr

A Study on Software Reliability using kappa(2) Distribution

Hee-Cheul Kim*, Sang-Sik Lee**
*Dept of Industrial Engineering, Nam-Seoul University
**Dept of Medical Information System, Songho College

요 약

본 논문에서는 순서 통계량을 이용한 유한 고장 NHPP 모형들이 제안되었다. 이 모형들은 결함당 고장발생률이 단조 증가하거나 단조 감소하는 패턴을 가진다. 그리고 수명 분포에서는 기존의 모형들과 비교하고 kappa(2) 분포를 이용한 소프트웨어 신뢰성 모형을 제안하여 이 모형의 특성과 효율성에 대하여 제시하였다. 고장 간격 시간 자료를 이용한 무한고장 NHPP 모형들에 대한 모수 추정은 최우 추정법을 사용하였고 적용 분포들의 적용을 용이하게 하기 위하여 특수한 형태를 제시하였다. 실제고장 자료를 이용한 자료분석에서는 편차자승합을 이용한 모형 선택과 적합도 및 치우침 검정을 실시하여 그 결과를 나열 하였다.

1. 서론

신뢰도에서 관측시간 $(0, t]$사이에 발견된 고장수 $N(t)$를 모형화 하는데 비동질적 포아송 과정(NHPP)이 널리 사용하여 왔다. 이 과정(Process)에서 강도함수(intensity function) 혹은 고장 발생률(Rate of occurrence of failure; ROCOF) $\lambda(t)=d\,E[N(t)]/dt$은 t에 대한 단조(Monotonic)함수로 흔히 가정한다.

이 범주에서 지금까지 알려진 모형들은 Goel-Okumoto 모형, Weibull 모형 그리고 Cox-Lewis 모형 등이 있는데 이 모형들에 대한 강도함수는 각각 시간에 관한 부분(Fraction) 함수, 멱(Power) 함수, 대수선형(Log-linear) 함수를 가정하였다[5].

NHPP 모형에서 평균값 함수 $m(t)$(Mean value function)와 강도 함수 $\lambda(t)$는 다음과 같은 관계로 표현 할 수 있다[2].

$$m(t)=\int_0^t \lambda(s)ds, \quad \frac{dm(t)}{dt}=\lambda(t) \qquad (1)$$

따라서 $N(t)$는 모수 $m(t)$를 가진 포아송 확률밀도함수(Probality density function)로 알려져 있다. 즉,

$$P\{N(t)=n\}=\frac{[m(t)]^n \cdot e^{-m(t)}}{n!}, \quad n=0,1,2,\ldots\infty \qquad (2)$$

이처럼 시간관련 모형(Time domain models)들은 NHPP에 의해서 확률 고장 과정으로 설명이 가능하다. 이러한 모형들은 고장 강도 함수 $\lambda(t)$가 다르게 표현됨으로서 평균값 함수 $m(t)$도 역시 다르게 나타난다. 이러한 NHPP 모형들은 유한 고장 모형과 무한 고장 범주로 분류한다[4]. 유한 고장 NHPP 모형들은 충분한 테스트 시간이 주어지면 결함들(Faults)의 기대값이 유한 값 ($\lim_{t\to\infty} m(t)=\theta<\infty$)을 가지고 반면에 무한 고장 NHPP 모형들은 무한 값을 가진다고 가정 된다

즉, 고장 시간을 일반화된 순서 통계량(Generalized order statistics;GOS) 모형을 사용하면 유한 고장 모형이 되고 기록값 통계량(Recore value statistics ; RVS)모형을 사용하면 무한 고장 모형이 됨이 알려져 있다[4]. 일반적으로 GOS를 이용한 모형은 소프트웨어 테스팅 시점에서 미지의 N 개의 결함들(Faults)을 가지고 있고 이 N 개 결함들로부터 임의의 확률밀도함수(Pdf)에 따라 발생된 n 개의 순서통계량이 고장 시점이 된다. 이 모형은 각 수리 시점에서 새로운 결함이 발생하지 않는다는 가정을 한다. 그러나 실제 상황에서는 수리시점에서도 고장이 발생 할 수도 있다. 이러한 상황을 추가하기 위하여 기록 멈춤 통계량(Record breaking statistics)을 사용하는

RVS 모형을 사용 할 수 있다[4].

유한 고장 NHPP 모형에서 충분한 테스트 시간이 주어졌을 때 탐색되어 질 수 있는 결함의 기대값을 θ라고 표현하고 $F(t)$를 분포함수라고 표현하면 유한 고장 NHPP모형의 평균값 함수는 다음과 같이 표현 할 수 있다[2].

$$m(t) = \theta F(t) \tag{3}$$

(3)식으로 부터 순간고장 강도함수(Instantaneous failure intensity) $\lambda(t)$는 다음과 같이 유도된다.

$$\lambda(t) = \theta F'(t) = \theta f(x) \tag{4}$$

(5) 식을 다음과 같이 변형하여 표기 할 수도 있다.

$$\lambda(t) = [\,\theta - m(t)][\,F'(t)/(1-F(t))] = [\,\theta - m(t)]\,h(t) \tag{5}$$

단, $h(t) = f(t)/(1-F(t))$는 위험함수(Hazard function, 고장률 함수)로서 소프트웨어 결함당 고장 발생률을 의미하고 $[\,\theta - m(t)]$은 t 시점에서 소프트웨어에 남아있는 결함들의 기대값을 나타낸다.

$[\,\theta - m(t)]$의 값은 시점 t에 대한 단조 비증가 함수(Monotonically non-increasing function)가 된다. 즉, 시간이 지남에 따라 결함들이 탐색되어 제거되기 때문에 감소성을 가진다. 따라서 $\lambda(t)$는 $h(t)$의 값에 따라 달라지며 상수, 증가, 감소 혹은 증가하다가 감소하는 패턴을 가질 수 있다. 이 분야의 기본적 모형인 Goel-Okumoto 모형은 $h(t)$가 정수 패턴을 가짐으로서 시점 t 에 독립이고 잘 알려진 S 모형(Yamada, Ohba-Osaki 모형)은 증가패턴을 가진다[2, 9]. 지금까지 신뢰성분야의 모형들은 대부분 감마족 분포(와이블, 지수, 감마분포)를 이용한 신뢰성 모형을 제안하여 연구되었고 또, 위험함수가 증가하다가 감소하는 비감족 분포인 로그 로지스틱 분포를 이용한 모형도 제안되었다[2]. 따라서 본 연구는 보다 많은 분포 정보가 함축되어 있고 의료정보분야 및 여러 분야에서 널리 사용되는 2모수 카파(kappa(2)) 분포[3,6]을 도입하여 신뢰성 모형을 제안하고자 한다.

2. 2모수 카파(kappa(2)) 성장 모형

의료정보분야 및 여러 분야에서 널리 사용되는 2모수 카파(kappa(2)) 분포의 확률밀도함수와 누적분포함수는 다음과 같다[3].

$$f_{kappa(2)}(t) = \alpha/\beta\,[\,\alpha + (t/\beta)^{\alpha}]^{-(\alpha+1)/\alpha} \tag{6}$$

$$F_{kappa(2)}(t) = [\,(\alpha/\beta)^{\alpha}/(\alpha + (t/\beta)^{\alpha})]^{1/\alpha} \tag{7}$$

단, 시간자료 는 $t(>0)$이고 $\beta(>0)$는 척도모수이고 $\alpha(>0)$는 형상모수이다.

(6),(7)을 유한 고장 NHPP 모형에 적용시킨 평균값 함수와 강도함수는 다음과 같다.

$$m(t) = \theta F_{kappa(2)}(t) \tag{8}$$

$$\lambda(t) = \theta F'(t) = \theta f_{kappa(2)}(x) \tag{9}$$

3. 최우추정법을 이용한 모수추정

시간 $(0, t]$까지 조사하기 위한 시간 절단(Time truncated)모형은 n번째 까지 고장시점 자료를

$$x_k = \sum_{i=1}^{k} t_k \;\; (k = 1, 2, \cdots, n\,;\, 0 \le x_1 \le x_2 \le \cdots \le x_n) \tag{10}$$

이라고 하면 데이터 집합 D_t는 $\{n, x_1, x_2 \cdots x_n; t\}$와 같이 구성된다. n번째까지 고장시점이 관찰된 고장 절단 모형일 경우에 데이터 집합 D_{x_n}은 $\{x_1, x_2 \cdots x_n\}$으로 구성된다. 이 시간 절단 모형에서의 우도함수는 다음과 같이 알려져 있다[4].

$$L_{NHPP}(\Theta \mid D_t) = \left(\prod_{i=1}^{n} \lambda(x_i)\right) \exp(-m(t)) \tag{11}$$

단, Θ은 미지의 모수 집합을 의미하고 우도함수 (7) 식에서 t을 x_n으로 대치하면 유사한 형태의 고장 절단 모형의 우도함수가 된다. (11)식과 (8), (9)식을 연관하면 유한 NHPP 모형에 대한 우도함수는 다음과 같은 형태로 표현 할 수 있다.

$$L_{NHPP_{FIN}}(\theta, \Theta \mid D_t) = \left(\prod_{i=1}^{n} \theta f(x_i)\right) \exp(-\theta F(x_n)) \tag{12}$$

따라서 최우추정법(MLE)을 이용하기 위한 $Gamma(\alpha, \beta)$ 모형 로그우도함수(Log likelihood function)는 다음과 같이 유도된다.

$$\ln L(\theta, \alpha, \beta \mid D_{x_n}) = n \ln \theta - n \log \Gamma(\alpha) + n\alpha \ln \beta + (\alpha - 1)\sum_{i=1}^{n} \ln x_i - \beta \sum_{j=1}^{n} x_i - \theta + \theta e^{-\beta x_n}\left(\sum_{j=0}^{\alpha-1} \frac{(\beta s_n)^j}{j!}\right) \tag{13}$$

형상 모수 α값은 상수(알고 있다고)라고 가정 했을 때 최우추정법을 이용하기 위하여 (13)식을 θ와 β에 대하여 편미분을 하면 다음과 같은 식을 유도 할 수 있다.

(14)

$$\frac{\alpha n}{\beta} = \sum_{i=1}^{n} x_i - \frac{\partial\left[\theta \exp(-\beta x_n) \sum_{i=0}^{\alpha-1} \frac{(\beta x_n)^i}{i!}\right]}{\partial \beta} \tag{15}$$

(14) 식과 (15) 식을 비선형 연립방정식(수치해석적 방법)을 이용하여 풀면 최우추정치 $\hat{\beta}_{MLE}$와 $\hat{\theta}_{MLE}$ 의 값을 구할 수 있다.
(14)과 (15) 식에서 $\alpha=1$이면 Goel-Okumoto 모형이되고 $\alpha=2$은 Yamada, Ohba-Osaki 모형이 된다.
유사한 방법으로 2모수 카파(kappa(2)) 성장 모형에 대한 로그우도함수는 다음과 같이 표현된다[3].

$$\ln L(\theta,\alpha,\beta|D_{x_n}) = n\ln\theta + n\ln\frac{\alpha}{\beta} - \left(\frac{\alpha+1}{\alpha}\right)\sum_{i=1}^{n}\ln(\alpha+(x_i/\beta)^{\alpha}) - \theta\left(\frac{(\alpha/\beta)^{\alpha}}{(\alpha+(x_n/\beta)^{\alpha}}\right)^{1/\alpha} \tag{16}$$

본 연구에서는 $\alpha=2$인 경우를 고려하면 즉, 형상모수 $\alpha=2$ 값은 상수(알고 있다고)라고 가정 했을 때 최우추정법을 이용하기 위하여 (16)식을 θ와 β에 대하여 편미분을 하면 다음과 같은 식을 유도 할 수 있다.

$$\frac{n}{\theta} = \left(\frac{(2/\beta)^2}{(2+(x_n/\beta)^2}\right)^{1/2} \tag{17}$$

$$\frac{2n}{\beta} = 3/2\sum_{i=1}^{n}\left(\frac{4\beta}{2\beta^2+x_i^2}\right) + \theta\left(\frac{4\beta}{\sqrt{2\beta^2+x_i^2}}\right) \tag{18}$$

(17) 식과 (18) 식을 비선형 연립방정식(수치해석적 방법)을 이용하여 풀면 최우추정치 $\hat{\beta}_{MLE}$와 $\hat{\theta}_{MLE}$ 의 값을 구할 수 있다.
NHPP 모형에서 테스트 시점 x_n에서 소프트웨어 고장이 일어난다고 하는 가정하에서 신뢰구간 $(x_n, x_n+t]$(단, t는 임무시간(Mission time))사이에서 소프트웨어의 고장이 일어나지 않을 확률인 신뢰도(Reliability) $\hat{R}(t\,|\,x_n)$는 다음과 같이 됨이 알려져 있다[8].

$$\hat{R}(t\,|\,x_n) = e^{-\int_{x_n}^{x_n+t}\lambda(\tau)d\tau} = \exp[-\{m(t+x_n)-m(x_n)\}]$$

3. 소프트웨어 고장 자료 분석

본 절에서는 86번의 고장 간격자료인 SYS2 (Allen P.Nikora and Michael R.Lyu)[7](고장시간 $\times 10^{-3}$ 만큼 이동 시킨 자료)를 이용하여 분석하고자 한다. 소프트웨어 신뢰성 모형의 모수 추정은 최우추정법을 이용하였고 비선형 방정식의 계산방법은 수치해석적 기본 방법인 이분법(Bisection method)을 사용하였다. 이러한 계산은 초기값을 10^{-6}와 10을, 허용 한계(Tolerance for width of interval)는 10^{-10}을 주고 수렴성을 확인 하면서 충분한 반복 횟수인 100번을 C-언어를 이용하여 모수 추정을 수행한 모수의 추정값들의 결과는 <표 1>에 요약되었다.

<표 1> 각 모형의 모수 추정값

모형	$\hat{\beta}_{MLE}$	$\hat{\theta}_{MLE}$
Goel-Okumoto Model	0.059195	84.87
Yamada-Ohba-Osaki Model	0.019679	85.43
kappa Model($\alpha=2$)	27.98765	89.78

모형 선택의 하나의 방법으로 편차자승합(C_{SSE})[8]을 이용할 수 있는데 이 편차자승합이 작으면 상대적으로 효율적인 모형이 된다. 주어진 자료를 이용하여 제시된 모형들에 대한 편차자승합의 값은 <표 2>에 요약되었다. 이 표에서 kapp(2) 모형이 이 분야에서 기존에 알려진 모형인 Yamada-Ohba-Osaki 모형이나 Goel-Okumoto 모형에 비해 상대적으로 효율적 모형으로 나타나고 있다.

참고문헌

[1] Goel, A. L. and K. Okumoto. “Time-Dependent Error-Detection Rate Models for Software Reliability and Other Performance Measures". *IEEE Trans. on Reliability*, R-28(3):206-211, Aug. 1979.

[2] Gokhale, S. S. and Trivedi, K. S. "A time/structure based software reliability model". *Annals of Software Engineering*. 8, 85-121. 1999

[3] Hosking, J. R. M. "The Four-parameter Kappa distribution". *IBM J. RES, DEVELOP, Vol. 38, No 3, May,* 1994.

[4] Kuo, L. and Yang, T. Y. "Bayesian Computation of Software Reliability". *Journal of the American Statistical Association*, Vol.91, pp.763-773, 1996.

[5] Lawless, J. F. “*Statistical Models and Methods for Lifetime Data*”. John Wiley & Sons, New York, 1981.

[6] Mielke, M. I. “Another Family Distribution for Describing and Analying Precipitation Data", *J. Appl. Meteorol* 12, 275-280, 1973.

[7] Nikora, A. P. and Lyu, M. R. *Handbook of Software Reliability Engineering, MR Lyu, Editor, chapter Software Reliability Measurement Experience*, pp.255-301. MacGraw-Hill, New York, 1996.

[8] Pham, H. and Nordmann, L. and Zhang, X. "A GeneralImperfect-Software-Debugging Model with S-Shaped Fault-Detection Rate". *IEEE Trans. on reliability*, VOL, 48, NO 2, 1999.

[9] Yamada, S. Ohba, M. and Osaki, S. "S-Shaped Reliability Growth Modeling for software Error Detection". *IEEE Trans. on Reliability*. R-32(5):475-485, Dec. 1983.

<표 2> 각 모형에 대한 편차자승합의 값

i	x_i	n_i	expontial model	$(n_i - \hat{m}(x_i))^2$	Yamada,Ohba-Osaki model	$(n_i - \hat{m}(x_i))^2$	kappa model	$(n_i - \hat{m}(x_i))^2$
1	0.479	1	2.372737018	1.884406922	0.813218499	0.034887329	1.086430164	0.007470173
2	0.745	2	3.661607907	2.760940835	1.267980659	0.535852316	1.68957493	0.096363724
3	1.022	3	4.98238092	3.929834113	1.74386532	1.577874335	2.317417441	0.46591895
4	1.576	4	7.55983584	12.67243121	2.70244902	1.683638546	3.571990656	0.183191999
5	2.610	5	12.15010319	51.12397569	4.51420527	0.23599652	5.907400809	0.823376228
6	2.859	6	13.21415117	52.04397706	4.954581154	1.092900563	6.468181972	0.219194359
7	3.552	7	16.09428798	82.7060739	6.187844979	0.659595779	8.024708678	1.050027875
8	4.149	8	18.48245482	109.881859	7.258635618	0.549621147	9.359841794	1.849169704
9	4.266	9	18.94067761	98.81707128	7.469333154	2.342940993	9.620795456	0.385386998
10	4.436	10	19.60084125	92.17615267	7.775944574	4.946422537	9.999518014	2.3231E-07
11	4.553	11	20.05134506	81.92684741	7.987281331	9.07647378	10.25985585	0.547813357
12	5.827	12	24.76007869	162.8196081	10.30369436	2.877452827	13.07636628	1.158564365
13	6.296	13	26.40601868	179.7213369	11.16263613	3.37590599	14.10382063	1.218419984
14	7.470	14	30.33123434	266.7092149	13.3243879	0.456451709	16.65017335	7.023418789
15	8.163	15	32.52339936	307.0695253	14.60668873	0.154693756	18.13438272	9.824355063
16	10.071	16	38.11434675	489.0443321	18.15155917	4.629206871	22.13852204	37.68145287
17	10.206	17	38.48652446	461.6707334	18.40279761	1.967841125	22.41689863	29.34279073
18	10.483	18	39.24092886	451.177059	18.91834641	0.843360127	22.98592573	24.85945534
19	11.079	19	40.82278431	476.2339149	20.02756545	1.055890746	24.20019273	27.04200446
20	11.836	20	42.75314613	517.705659	21.43553358	2.060756653	25.72200888	32.74138558
21	12.273	21	43.82874996	521.1518249	22.24743359	1.556090553	26.58975383	31.24534784
22	14.503	22	48.90446499	723.8502366	26.37116756	19.10710581	30.88867522	79.00854714
23	14.940	23	49.82298567	719.4725604	27.1738915	17.42137024	31.70482614	75.77399808
24	15.280	24	50.52136931	703.3830302	27.79688046	14.41630123	32.33367906	69.45020663
25	15.685	25	51.33513052	693.5390993	28.53704728	12.51070349	33.07567577	65.21653917
26	16.220	26	52.38062248	695.9372425	29.51133506	12.32947389	34.04393841	64.70494514
27	16.497	27	52.90906278	671.2795341	30.01412239	9.084933782	34.53989685	56.85004455
28	16.860	28	53.58857879	654.7753647	30.67119261	7.135269938	35.18425583	51.61353184
29	17.382	29	54.54049052	652.3166562	31.61225341	6.82386787	36.09969794	50.40571085
30	17.995	30	55.62143621	656.4579936	32.71123357	7.350787473	37.15782432	51.23444907
31	18.272	31	56.09717013	629.8679488	33.2055362	4.864389951	37.62994882	43.9562213
32	19.572	32	58.22854179	687.9364042	35.50467146	12.28272202	39.79541644	60.76851746
33	20.393	33	59.49249551	701.8523185	36.93768648	15.50537478	41.1201788	65.93730376
34	20.606	34	59.81050252	666.1820404	37.30690183	10.9355997	41.45845824	55.62859938
35	22.226	35	62.10228915	734.5340773	40.07804672	25.78655846	43.95848625	80.25447592
36	23.827	36	64.1612296	793.0548527	42.74751849	45.52900579	46.3038184	106.1686736
37	24.125	37	64.52339089	757.5370462	43.23628565	38.89125874	46.72677208	94.61009507
38	24.999	38	65.54945637	758.9725464	44.65442289	44.281344	47.94295549	98.86236381
39	25.617	39	66.24360055	742.2137711	45.64298573	44.12925943	48.78125385	95.6729269
40	28.257	40	68.93888328	837.4589653	49.72649522	94.60470919	52.16547129	147.9986918
41	28.262	41	68.9435988	780.8447137	49.73400698	76.28287787	52.17158502	124.8043119
42	28.411	42	69.08348272	733.5150363	49.95746164	63.32119572	52.35327191	107.1902392
43	29.445	43	70.020963	730.1324415	51.48683969	72.02644797	53.58759617	112.0971926
44	31.886	44	72.0192211	785.0767512	54.94568805	119.8080868	56.3234584	151.867627
45	32.346	45	72.36451409	748.8166315	55.57309082	111.7902496	56.81203455	139.5241603
46	32.911	46	72.77595722	716.9518853	56.33294105	106.7696707	57.40082117	129.9787234
47	34.030	47	73.55131906	704.9725439	57.8025602	116.6953068	58.53094184	132.9626198
48	34.467	48	73.84044537	667.728617	58.36367531	107.4057659	58.95959202	120.1126572
49	35.394	49	74.42955401	646.6622172	59.53000616	110.8810296	59.84591802	117.6339377
50	39.856	50	76.85382992	721.1281814	64.68655926	215.695023	63.70382149	187.7947234
51	40.570	51	77.18571587	685.6917157	65.44162664	208.5605801	64.26287758	175.9039217
52	40.751	52	77.2676443	638.4538485	65.62999333	185.7767183	64.40221242	153.8148728
53	42.236	53	77.90768182	620.3926137	67.12925946	199.635973	65.51002228	156.5006573
54	42.993	54	78.21292969	586.2659641	67.86211291	192.1581742	66.0512443	145.2324891
55	46.147	55	79.34726272	592.7892018	70.69227322	246.2474388	68.14711971	172.8467566
56	48.262	56	79.99756367	575.8830621	72.39477428	268.7886236	69.41997592	180.0957538
57	49.146	57	80.24614757	540.3833768	73.06165653	257.9768106	69.92299868	167.0038949
58	51.183	58	80.77177577	518.5537716	74.50166295	272.3048801	71.02140819	169.5570711
59	52.664	59	81.11604499	489.1194461	75.46694361	271.1602318	71.76998415	163.0724953
60	53.223	60	81.23834134	451.0671427	75.81404107	250.0838951	72.04218891	145.0143138
61	53.713	61	81.34226376	413.807695	76.11070323	228.3333521	72.27629348	127.1547946
62	54.306	62	81.46406566	378.849852	76.46038989	209.1028758	72.55411139	111.3892673
63	56.075	63	81.80302068	353.5535869	77.44453513	208.6445952	73.34850412	107.0915375
64	56.160	64	81.81843073	317.4964736	77.4896533	181.9707462	73.38542141	88.08613504
65	58.996	65	82.29055239	298.963202	78.88667464	192.8397327	74.55542515	91.30614975
66	59.209	66	82.32291387	266.4375173	78.98340772	168.568876	74.63866538	74.6265395
67	61.075	67	82.58963337	243.0366687	79.78474147	163.4496145	75.34204211	69.58966665
68	61.565	68	82.65492381	214.7667919	79.98188581	143.5655876	75.51931821	56.54014633
69	63.052	69	82.84185295	191.5968932	80.54799691	133.3562326	76.03936064	49.5525982
70	67.374	70	83.30044376	176.9018042	81.94145137	142.5982608	77.41017216	54.91065139
71	68.792	71	83.42708594	154.4324649	82.32514385	128.2588833	77.8185654	46.49283416
72	69.815	72	83.51207048	132.5277668	82.58167188	111.9717798	78.10162341	37.22980827
73	75.305	73	83.88978305	118.587375	83.70315707	114.5575714	79.47129816	41.8776999
74	76.825	74	83.97442058	99.48906597	83.94753108	98.9533746	79.81044169	33.76123267
75	80.106	75	84.13311049	83.41370717	84.39433172	88.25346844	80.49067447	30.14750612
76	82.822	76	84.24305473	67.94795121	84.69212369	75.55301431	81.00502819	25.05030717
77	84.997	77	84.3192039	53.5707457	84.89080654	62.26482785	81.38819195	19.25622859
78	88.502	78	84.42304322	41.25548415	85.14872702	51.10429797	81.95713004	15.65887818
79	89.227	79	84.44196203	29.61495072	85.19376696	38.36274911	82.06789728	9.411993724
80	91.190	80	84.48929192	20.15374197	85.30332347	28.12523986	82.35676081	5.554321528
81	95.169	81	84.56989835	12.7441742	85.47783955	20.05104702	82.89647437	3.596615035
82	96.259	82	84.58886266	6.702209849	85.51628832	12.36428356	83.03445671	1.070100693
83	96.504	83	84.59295946	2.537519833	85.52444675	6.372831401	83.06492193	0.004214857
84	97.698	84	84.61209641	0.374662018	85.56182385	2.439293745	83.2105899	0.623168303
85	98.692	85	84.62702788	0.139108202	85.59010618	0.348225309	83.32840521	2.794229158
86	102.594	86	84.6778406	1.748105486	85.67985041	0.102495761	83.76254515	5.006204208
C_{SSE}			$\sum_{i=1}^{86} (n_i - \hat{m}(x_i))^2$	34518.92715	$\sum_{i=1}^{86} (n_i - \hat{m}(x_i))^2$	6549.586126	$\sum_{i=1}^{86} (n_i - \hat{m}(x_i))^2$	5682.672198