독립심화학습보고서

연구 주제 관련한 논문 요약 및 분석과 실질적 구현

-역 문제에서의 불확실성 정량화를 위한 표준오차 계산 : 점근적 이론과 부트스트래핑 방법의 비교-

지도교수 이 선 미

경희대학교 응용과학대학
응용수학 전공

정 상 만

- 목 차 -

- 선 정 논 문 -

Standard Error Computations for Uncertainty Quantification in Inverse Problems: Asymptotic Theory vs. Bootstrapping

H. T. Banks, Kathleen Holm, and Danielle Robbins
Center for Research in Scientific Computation and Center for Quantitative Sciences in Biomedicine
North Carolina State University Raleigh, NC 27695-821

Ⅰ. 논 문 요 약

1. 논문의 연구 목적

비선형 동역학계(ODE or integral equation)의 역문제(inverse problem)를 해결하는 과정에서 계산적으로(computationally) 효율적인 매개변수 추정을 위한 방법을 모색한다. 이 때, 비선형 동역학계는 매개변수에 의존하는 방정식이므로, 역문제에서 매개변수를 잘 추정하는 것은 중요한 문제이다. 구체적으로, 동역학계로 대표되는 미분방정식의 일반해를 비선형 회귀모형으로 하여 알려지지 않은 매개변수, 즉 모수를 추정하기 위해 비모수적 방법을 이용한다. 여기서 모수는, 회귀계수를 포함하여 오차 분석, 분포를 구하기 위한 표준편차 등을 일컫는다. 제한된 표본을 갖고 가정이 성립하는 경우와 성립하지 않는 경우에 대한 논문의 두 방법(부트스트랩, 점근적 이론)을 이용한 근삿값을 구하고, 이를 비교하여 두 방법의 장단점을 분석하고, 주어지는 상황에 따른 향상된 해결방안을 제시하고자 한다. 최종적으로는 두 방법에 대한 표준오차를 계산하여 불확실성을 정량화하는데 있어 효율적인 방법이 무엇인지 제시하고자 한다.

초록(abstract)의 내용을 요약하면 다음과 같다.

-동역학계에 종속된 비선형 매개변수에 관한 역문제의 불확실성 측정을 위한 전산적인 두 접근법으로 조사하고자 한다.

-noise forms & levels 와 같은 data와 연관된 문제에서의 부트스트래핑과 점근선 이론이라는 접근 방법을 비교할 것이다.

-부트스트래핑과 점근선 이론을 이용한 방법으로 매개변수 추정, 표준오차, 신뢰구간, 연산속도를 구하여 대조하고 비교할 것이다.

2. 논문에서 제시한 연구 문제 및 방법

2-1) 모형(Model) 설정

논문의 개요에 따르면, 매개변수에 의존하는 비선형 동역학계는 다음과 같은 형태를 갖는다.

$$\frac{dx}{dt}(t) = F(t, x(t), q), \quad x(0) = x_0$$

또한, 위 동역학계에 대한 관측값(observations)은 아래와 같은 형태를 갖는다.

$$y_j = f(t_j, \theta) = Cx(t_j; \theta), \ j = 1, \cdots\cdots, n$$

여기서 $x(t_j, \theta)$는 미지의 매개변수 q에 종속된 동역학계의 해이고, x_0은 초기조건으로, 구하고자 하는 모수의 $\theta = (q, x_0)$ 의 형태를 갖는다.

우리는 모수 θ를 추정하기 위해 θ의 추정량을 $\hat{\theta}$ 라 하고 이를 부트스트랩 방법과 점근적 이론을 이용한 방법으로 구하고자 한다. 두 방법으로 추정치를 구할 때 비교하기 적합한 비선형 동역학계 모델 중 하나로 인구모델인 다음의 Verhulst-Pearl growth model을 채택한다.

$$\frac{dx(t)}{dt} = rx(t)\left(1 - \frac{x(t)}{K}\right), \ x(0) = x_0.$$

이 때 K는 $t \to \infty$ 일 때 대응하는 환경수용능력, r은 인구성장률이다. 모수 $\theta = (K, r, x_0)$ 에 대한 인구모델의 해석적 일반해는 변수분리 방법을 이용하면 다음과 같이 구할 수 있다.

$$x(t) = f(t, \theta) = \frac{K}{1 + \left(\frac{K}{x_0} - 1\right)e^{-rt}}, \ \theta = (K, r, x_0).$$

앞서 채택한 미분방정식이 로지스틱(Logistic) 모형이므로, 그 일반해 또한 로지스틱 곡선으로 표현된다. 우리는 예측하고자 하는 최종적인 참값인 모수를 $K=17.5$, $r=0.7$, $x_0=0.1$ 로 설정하고 이를 추정할 것이다.

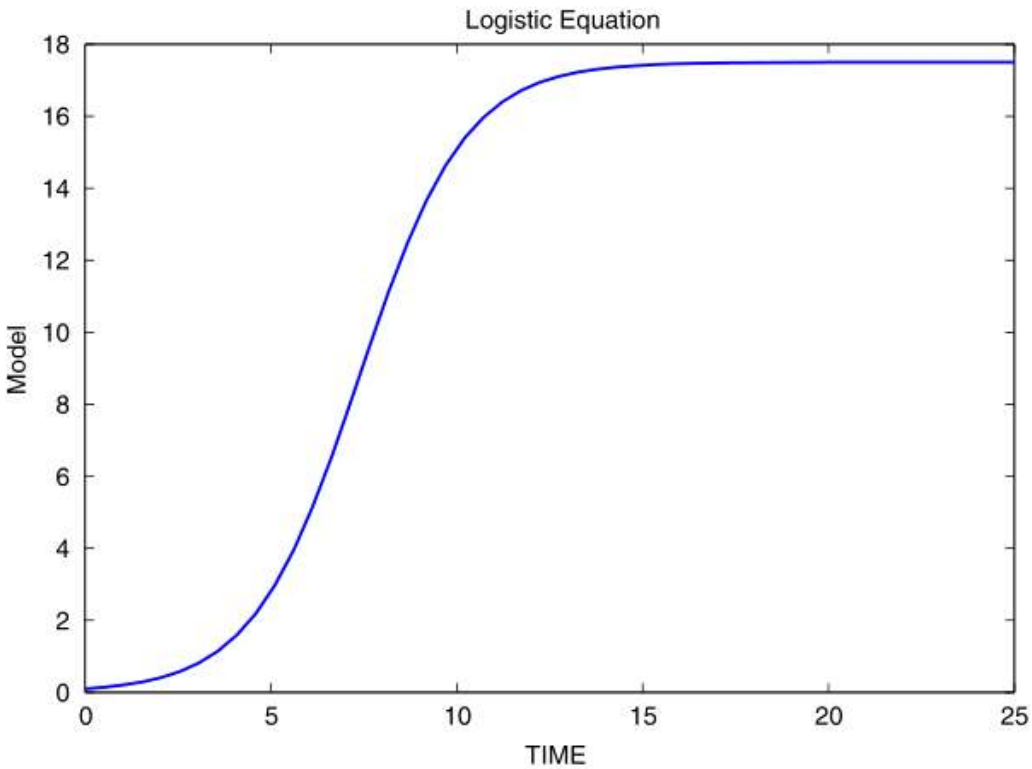


Fig. 1. Logistic curve with $K = 17.5, r = 0.7$ and $x_0 = 0.1$.

2-2) 등분산(Constant Variance Data)을 가정한 부트스트래핑

주어진 관측치 $(y_1,t_1),(y_2,t_2),\cdots\cdots,(y_n,t_n)$ 에 대하여 다음을 가정한다.

$$Y_j = f(t_j,\theta_0)+\varepsilon_j \ , \ j=1,\cdots\cdots,n$$

이 때, ε_j 는 독립동일분포(iid)로써, 평균($E(\varepsilon_j)=0$)을 0이고 분산을 σ^2 으로 갖는 F 분포로부터의 오차항이고 θ_0는 참값을 갖는 모수로 가정한다.
그리고 이에 대응하는 확률변수 $Y=\{Y_j\}$에 대한 실제값(realizations) $\{y_j\}$는 $y_j = f(t_j,\theta_0)+\epsilon_j$ 로 주어진다고 하자. 이제 오차를 추정하기 위해 일반 최소제곱 추정량을 계산하고자 한다. 아래와 같은 식을 사용하여 θ에 대한 OLS 방법을 적용한다.

$$\theta_{OLS}(Y)=\theta^n_{OLS}(Y)=\mathrm{argmin}_{\theta\in\Theta_{ad}}\sum_{j=1}^{n}[\,Y_j-f(t_j,\theta)]^2$$

여기서 $\Theta_{ad}\subset R^p$는 적용 가능한 모수 집합이다. θ_{OLS} 모델과 관측값(observations)과의 오차를 최소화하기 위해 모델에 대응하는 다음 식

$$\sum_{j=1}^{n}[\,Y_j-f(t_j,\theta)]\bigtriangledown f(t_j,\theta)=0$$

을 θ에 관하여 풀면, θ_{OLS}는 확률변수이므로, 랜덤 프로세스 $\{Y_j\}_{j=1}^n$ 의 실제값이 $\{y_j\}_{j=1}^n$ 일 때 θ의 OLS추정량

$$\hat{\theta}_{OLS}=\hat{\theta}^n_{OLS}=\mathrm{argmin}_{\theta\in\Theta_{ad}}\sum_{j=1}^{n}[\,Y_j-f(t_j,\theta)]^2$$

을 구할 수 있고, 최종적으로 θ_{OLS}의 추정치 $\hat{\theta}_{OLS}$ 를 얻는다. 이 때, 분산 σ^2는

$$\sigma_0^2=\frac{1}{n}E\left[\sum_{j=1}^{n}[\,Y_j-f(t_j,\theta_0)]^2\right]$$

이므로 앞서 구한 $\hat{\theta}_{OLS}$ 를 이용하면 모분산의 OLS 추정량을 얻을 수 있다. 이에 따른 불편추정량은 다음과 같이 주어진다.

$$\hat{\sigma}^2_{OLS} = \frac{1}{n-p}\sum_{j=1}^{n}(y_j - f(t_j,\hat{\theta}))^2$$

이제 부트스트래핑 추정량 $\hat{\theta}_{boot}$ 를 계산하여, 실제적인 분포를 얻기 위해 다음의 알고리즘을 적용한다.

1. OLS 방법을 적용하여 각 $\{y_j\}_{j=1}^{n}$ 로부터 추정량 $\hat{\theta}^0 = (\hat{K}^0, \hat{r}^0, \hat{x}_0^0)$을 계산한다.

2. 1에서의 추정량을 이용하여 표준화된 잔차 $\overline{r_j}$를 다음과 같이 정의한다.

$$\overline{r_j} = \sqrt{\frac{n}{n-p}}\left(y_j - f(t_j, \hat{\theta}^0)\right) \text{ for } j = 1,\cdots,n.$$

그러면 $\{\overline{r_1},\cdots,\overline{r_n}\}$은 $i.i.d$ 한 실제적인 분포 F_n의 확률변수 $\overline{R}_j$ 의 실제값이 되고, 이 때 모수의 개수 $p=3$ 이다. 평균(기댓값)과 분산은 다음과 같다. m은 $m=0$ 으로 설정한다.

$$E(\overline{R}_j | F_n) = n^{-1}\sum_{j=1}^{n}\overline{r}_j = 0 \ , \ Var(\overline{R}_j | F_n) = n^{-1}\sum_{j=1}^{n}\overline{r}_j^2 = \hat{\sigma}^2.$$

3. 데이터 $\{\overline{r_1},\cdots,\overline{r_n}\}$로부터 무작위복원추출을 하여 크기가 n인 부트스트랩 표본 $\{r_1^m,\cdots,r_n^m\}$을 생성한다.

4. 다음과 같은 부트스트랩 표본 포인트(bootstrap sample points)를 생성한다.

$$y_j^m = f(t_j, \hat{\theta}^0) + r_j^m \ , \text{ where } j = 1,\cdots,n.$$

5. OLS 방법을 이용하여 부트스트랩 표본 $\{y_j^m\}$로부터 새로운 추정량 $\hat{\theta}^{m+1} = (\hat{K}^{m+1}, \hat{r}^{m+1}, \hat{x_0}^{m+1})$을 구한다. 그 후에 $\hat{\theta}^{m+1}$를 부트스트랩 추정량들을 갖는 길이가 M인(이 예제의 경우는 $3M$을 사용) 벡터 Θ에 추가시킨다.

6. 이제 $m = m+1$로 두고, 3-5 까지의 과정을 반복한다.

7. 임의의 큰 M (여기선, $M=1000$)에 대해 위의 반복과정을 M번 수행하면, 길이가 M인 벡터 Θ를 얻을 수 있다.

8. 1-7의 과정을 거친 후, Θ로부터 다음과 같은 공식을 이용하여 부트스트랩 평균, 표준오차, 신뢰구간을 계산한다.

$$\hat{\theta}_{boot} = \frac{1}{M}\sum_{m=1}^{M}\hat{\theta}^m \ ,$$

$$Cov(\hat{\theta}_{boot}) = \frac{1}{M-1}\sum_{m=1}^{M}(\hat{\theta}^m - \hat{\theta}_{boot})^T(\hat{\theta}^m - \hat{\theta}_{boot}) \ ,$$

$$SE_k(\hat{\theta}_{boot}) = \sqrt{Cov(\hat{\theta}_{boot})_{kk}} \ .$$

2-3) 등분산이 아닌 경우(Non-Constant Variance Data)의 부트스트래핑

주어진 관측치 $(y_1,t_1),(y_2,t_2),\cdots\cdots,(y_n,t_n)$ 에 대하여 다음을 가정한다.

$$Y_j = f(t_j,\theta_0)(1+\varepsilon_j)\ ,\ j=1,\cdots\cdots,n$$

이 때, ε_j 는 독립동일분포(iid)로써, 평균($E(\varepsilon_j)=0$)을 0이고 등분산이 아닌 F 분포로부터의 오차항이다. 또한 평균 $E(Y_j)=f(t_j,\theta_0)$, 분산 $Var(Y_j)=\sigma_0^2 f^2(t_j,\theta_0)$ 를 갖고, 이에 대응하는 확률변수 $Y=\{Y_j\}$에 대한 실제값(realizations) $\{y_j\}$는

$$y_j = f(t_j,\theta_0)(1+\epsilon_j)$$

로 주어진다고 하자. 이러한 관점에서 생성된 분산은 모형에 의존하고, 그런 이유로 수직적으로 등분산이 아님을 확인할 수 있다. θ_0와 σ_0^2을 추정하기 위해 GLS(Generalized least square) 의 특수한 형태를 이용하는 방법을 채택할 것이다.

확률변수 θ_{GLS}를 정의하기 위해 다음의 방정식을 추정량 θ_{GLS}에 대하여 푼다.

$$\sum_{j=1}^{n} w_j[\,Y_j - f(t_j,\theta_{GLS})]\bigtriangledown f(t_j,\hat{\theta}) = 0\ ,\ \text{where}\ \ w_j = f^{-2}(t_j,\theta_{GLS}).$$

이러한 방정식은 정규방정식(normal equation)이라 하고, θ에 의존하지 않는 가중치 w_j로 $\bigtriangledown f$의 가중최소제곱판정을 이용하여 0이 되도록 하는 대수적 조작에 포함된다. 양 θ_{GLS}는 확률변수이므로, $\{y_j\}_{j=1}^{n}$ 이 랜덤 프로세스 Y_j의 실제값(realization)이면, $\hat{\theta}$에 관하여 다음을 풀어 θ_{GLS}의 추정량 $\hat{\theta}_{GLS}$ 를 얻는다.

$$\sum_{j=1}^{n} f^{-2}(t_j,(\widehat{\theta}))\left[y_j - f(t_j,\hat{\theta})\right]\bigtriangledown f(t_j,\hat{\theta}) = 0$$

추정량 $\hat{\theta}_{GLS}$는 반복적으로 풀어질 수 있다. 그러한 반복과정은 다음과 같이 요약할 수 있다.

1. OLS 방정식 $\hat{\theta}_{OLS}=\hat{\theta}^{n}_{OLS}=\arg min_{\theta\in\Theta_{ad}}\sum_{j=1}^{n}[Y_j-f(t_j,\theta)]^2$를 이용하여 $\hat{\theta}^{(0)}$의 $\hat{\theta}_{GLS}$를 추정한다.

2. 가중치 $\hat{w}_j=f^{-2}(t_j,\hat{\theta}^{(k)})$를 구한다.

3. $\hat{\theta}^{(k+1)}=\arg min_{\theta\in\Theta_{ad}}\sum_{j=1}^{n}\hat{w}_j(y_j-f(t_j,\theta))^2$ 를 풀어서 $\hat{\theta}_{GLS}$의 $k+1$번째 추정량 $\hat{\theta}^{(k+1)}$ 를 얻기 위해 $\hat{\theta}$를 재-추정(Re-estimate)한다.

4. $k=k+1$으로 두고 항목 2로 돌아간다. $\hat{\theta}_{GLS}$의 연속적인 추정치 중 두 개가 충분히 가까워질 경우 이 과정을 종료한다.

참고문헌으로 제시한 문헌에서의 표준 알고리즘은 θ_0의 부트스트랩 추정량 $\hat{\theta}_{boot}$과 실제적인 분포를 계산하기 위해 사용될 수 있다. 여기서 우리는, 모수가 θ인 모형 출력값의 비선형 종속성의 일반적 경우로 다루도록 한다.

1. GLS 방법을 적용하여 각 $\{y_j\}_{j=1}^{n}$ 로부터 추정량 $\hat{\theta}^0=(\hat{K}^0,\hat{r}^0,\hat{x}_0^0)$을 계산한다.

2. NCV인 표준화된 잔차(residuals)를 다음과 같이 정의한다.

$$\bar{s}_j=\frac{y_j-f(t_j,\hat{\theta}^0)}{f(t_j,\hat{\theta}^0)}$$

그러면 $\{\bar{s}_1,\cdots,\bar{s}_n\}$는 실제적인 분포(empirical distribution) F_n를 갖는 $i.i.d.$인 확률변수 $\bar{S}_j$의 실제값이 된다. 여기서 이러한 비선형 가중치의 경우에 원하는 평

균과 분산의 조건들은 오직 다음과 같은 근사식에서만 성립된다. 이 때, m은 $m=0$으로 둔다.

$$E(\overline{S}_j|F_n = n^{-1}\sum_{j=1}^{n}\overline{s}_j \approx 0,\ \ Var(\overline{S}_j|F_n) = n^{-1}\sum_{j=1}^{n}\overline{s}_j^{\,2} \approx \widehat{\sigma^2}.$$

3. 데이터 $\{\overline{s_1},\cdots,\overline{s_n}\}$로부터 무작위복원추출을 하여 크기가 n인 부트스트랩 표본 $\{s_1^m,\cdots,s_n^m\}$을 생성한다.

4. 다음과 같은 부트스트랩 표본 포인트(bootstrap sample points)를 생성한다.

$$y_j^m = f(t_j,\hat{\theta}^0) + r_j^m \ \ , \text{ where } \ j=1,\cdots,n.$$

5. GLS방법을 이용하여 부트스트랩 표본 $\{y_j^m\}$로부터 새로운 추정량 $\hat{\theta}^{m+1} = (\widehat{K}^{m+1}, \hat{r}^{m+1}, \widehat{x_0}^{m+1})$을 구한다. 그 후에 $\hat{\theta}^{m+1}$를 부트스트랩 추정량들을 갖는 길이가 M인(이 예제의 경우는 $3M$을 사용) 벡터 Θ에 추가시킨다.

6. 이제 $m=m+1$로 두고, 3-5번의 과정을 반복한다.

7. 임의의 큰 M (여기선, $M=1000$)에 대해 위의 반복과정을 M번 수행하면, 길이가 M인 벡터 Θ를 얻을 수 있다.

8. 이제 이전의 식 $Cov(\hat{\theta}_{boot}) = \frac{1}{M-1}\sum_{m=1}^{M}(\hat{\theta}^m - \hat{\theta}_{boot})^T(\hat{\theta}^m = \hat{\theta}_{boot})$을 사용하여 Θ로부터 평균, 표준오차, 신뢰구간을 계산한다.

2-4) 등분산(Constant Variance Data)을 가정한 점근적 이론

앞서 설명된 주어진 통계적 모형과 실제값(realizations)에서, 점근적 이론을 이용하여 추정량들을 계산할 수 있다. 계산 알고리즘은 아래와 같이 주어진다.

1. 추정량 $\hat{\theta} = (\hat{K}, \hat{r}, \hat{x})$을 계산한다.

2. 민감도 행렬(sensitivity matrix) $\chi = \frac{\partial f}{\partial \theta}$ 와 분산추정량 $\hat{\sigma}^2$ 를 다음과 같이 계산한다. 주어진 로지스틱 모델은 다음과 같은 미분방정식의 관점으로 서술할 수 있다.

$$\frac{d\chi(t)}{dt} = F(\chi(t,\theta),\theta) = r\chi(t)\left(1 - \frac{\chi(t)}{K}\right) \text{ with } \frac{d}{dt}\frac{\partial \chi}{\partial \theta} = \frac{\partial F}{\partial \chi}\frac{\partial \chi}{\partial \theta} + \frac{\partial F}{\partial \theta} \text{ (sensitivity eq.)}$$

위 상미분방정식의 해를 구하면 다음의 민감도 행렬을 얻을 수 있다.

$$\chi_{j,k} = \frac{\partial \chi(t_j)}{\partial \theta_k} = \frac{\partial f(t_j,\theta)}{\partial \theta_k} \text{ , for } j = 1,\cdots,n,\ k = 1,\cdots,p \text{ , } \chi = \chi^n \text{ is an } n \times p \text{ matrix.}$$

등분산 가정 불편추정량은, 이전과 같이 다음에 의해 얻어진다.

$$\sigma_0^2 \approx \hat{\sigma}_{OLS}^2 = \frac{1}{n-p}\sum_{j=1}^{n}(y_j - f(t_j,\hat{\theta}))^2 \ .$$

3. 공분산 행렬(covariance matrix)을 추정한다.
참값 모수인 θ_0와 분산 σ_0^2 이 알려지지 않았으므로, 근사 참값 공분산 행렬 $\Sigma_0^n = \sigma_0^2\left[\chi^T(\theta_0)\chi(\theta_0)\right]^{-1}$ 은 알려져 있지 않다. 이는 특정한 조건하에서, 다음과 같이 θ의 추정량이 점근적으로 정규적(asymptotically normal)임을 보일 수 있다.

$$\hat{\theta}_n \sim N_p(\theta_0, \sigma_0^2\left[\chi^T(\theta_0)\chi(\theta_0)\right]^{-1})$$

우리는 $\hat{\sigma}^2_{OLS}$와 $\hat{\theta}$를 이용하여 공분산 행렬을 다음과 같이 계산한다.

$$\Sigma_0^n \approx \hat{\Sigma}^n(\hat{\theta}) = \sigma^2_{OLS}\left[\chi^T(\hat{\theta})\chi(\hat{\theta})\right]^{-1}$$

4. $\hat{\Sigma}^n(\hat{\theta})$를 이용하여 표준오차를 다음과 같이 계산한다.

$$SE_k(\hat{\theta}) = \sqrt{\hat{\Sigma}^n_{kk}(\hat{\theta})}$$

2-5) 등분산(Non-Constant Variance Data)이 아닌 경우 점근적 이론

주어진 관측치 $(y_1,t_1),(y_2,t_2),\cdots\cdots,(y_n,t_n)$ 에 대하여 다음을 가정한다.

$$Y_j = f(t_j,\theta_0)(1+\varepsilon_j) \ , \ j=1,\cdots\cdots,n$$

이 때, ε_j 는 독립동일분포(iid)로써, 평균($E(\varepsilon_j)=0$)을 0이고 등분산이 아닌 F 분포로부터의 오차항이다. 또한 평균 $E(Y_j)=f(t_j,\theta_0)$, 분산 $Var(Y_j)=\sigma_0^2 f^2(t_j,\theta_0)$ 를 갖고, 이에 대응하는 확률변수 $Y=\{Y_j\}$에 대한 실제값 $\{y_j\}$는

$$y_j = f(t_j,\theta_0)(1+\epsilon_j)$$

로 주어진다고 하자. 이 때, 점근적 이론을 이용하면, GLS 알고리즘을 이용한 추정량 $\hat{\theta}$ 를 얻을 수 있다. 그러면 σ_0^2은 다음과 같이 근사할 수 있다.

$$\sigma_0^2 \approx \hat{\sigma}^2_{GLS} = \frac{1}{n-p}\sum_{j=1}^{n}\frac{1}{f^2(t_j,\hat{\theta})}(f(t_j,\hat{\theta})-y_j)^2 \ .$$

이제 $\hat{\sigma}^2_{OLS}$와 $\hat{\theta}$를 이용하여 공분산 행렬을 다음과 같이 계산한다.

$$\Sigma_0^n \approx \hat{\Sigma}^n(\hat{\theta}) = \sigma^2_{OLS}\left[\chi^T(\hat{\theta})W(\hat{\theta})\chi(\hat{\theta})\right]^{-1} \ , \ W^{-1}(\theta) = diag(f^2(t_1,\theta,\cdots,f^2(t_n,\theta)).$$

표준오차는 $\hat{\Sigma}^n(\hat{\theta})$을 이용하여 다음과 같이 계산한다.

$$SE_k(\hat{\theta}) = \sqrt{\hat{\Sigma}^n_{kk}(\hat{\theta})} \ .$$

3. 연구결과 및 해석

3-1) 잡음 데이터 셋(noisy data sets) 구현

실제적인 계산을 위해 설정했던 모델의 시뮬레이션을 이용하여 잡음 데이터 셋을 생성하였다. 시간(time) t는 길이가 $n=50$, $t=(0,\cdots,25)$인 벡터로 생성하였다. 참인 모수값 $\theta_0=(17.5,0.7,0.1)$ 전제하의 주어진 모델에서, MATLAB의 내장함수인 $ode45$를 이용하여 모델의 해 $f(t_j,\theta_0)$을 계산하였다. 잡음 레벨을 Nl로 갖는 길이가 n인 잡음 벡터(noise vector) ϵ는 분포가 $N(0,Nl^2)$인 난수 생성기능으로부터 값을 취하였다. 등분산 데이터 셋은 $y_j=f(t_j,\theta_0)+\epsilon_j$ 으로부터 얻어지고, 이와 마찬가지로 등분산이 아닌 경우는 $y_j=f(t_j,\theta_0)(1+\epsilon_j)$를 이용하여 구할 수 있다. 등분산과 등분산이 아닌 경우의 데이터 셋은 1%, 5%, 10% 인 잡음(Nl)의 경우에서 생성하였다.

3-2) OLS 방법을 이용한 등분산 가정 데이터

모수 추정 계산을 이끌어내기 위해 OLS 방법을 이용한 등분산(CV) 데이터를 사용하였다. 부트스트랩 추정량들은 $M=1000$이고 $\theta_0=(17.5,0.7,0.1)$인 경우에서 계산되었다. 참고문헌에 따르면, 신뢰구간을 계산하기 위해 추정량과 표준오차를 계산해야 하고, 이는 $M=1000$으로 두는 것이 적절하다고 사료된다.
표준오차 SE_k와 이와 관련된 신뢰구간 $\left[\hat{\theta}_k-1.96SE_k,\hat{\theta}_k+1.96SE_k\right]$는 Table 1-3에 다른 결과들과 함께 나열하였다. Fig.2에서 우리는 $Nl=0.05$인 경우의 계산된 값에 대한 실제적인 분포를 제시하였다. 나머지 Nl의 경우는 이와 매우 유사하므로 생략한다. 모수 추정량들과 표준오차는 다음에 소개될 결과들과 같이 부트스트래핑과 점근적 이론으로 비교할 만하다. 그러나 계산속도(Table 4) 에서는 부트스트랩 방법이 점근적 이론보다 2~3배 더 느린 계산 속도를 가진다. 이런 이유로, 점근적 방법의 접근은 부트스트래핑보다 좀 더 이점이 있는 방법이다.

Table 1

Asymtotic and bootstrap OLS estimates for CV data, M=0.01.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{asy}$	17.498576	0.002021	(17.494615, 17.502537)
$\hat{r}_{asy}$	0.700186	0.000553	(0.699103, 0.701270)
$(\hat{x_0})_{asy}$	0.100044	0.000407	(0.099247, 0.100841)
$\hat{K}_{boot}$	17.498464	0.001973	(17.494597, 17.502331)
$\hat{r}_{boot}$	0.700193	0.000548	(0.699118, 0.701268)
$(\hat{x_0})_{boot}$	0.100034	0.000399	(0.099252, 0.100815)

Table 2

Asymtotic and bootstrap OLS estimates for CV data, M=0.05.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{asy}$	17.486571	0.010269	(17.466444, 17.506699)
$\hat{r}_{asy}$	0.702352	0.002825	(0.696815, 0.707889)
$(\hat{x_0})_{asy}$	0.098757	0.002050	(0.0947386, 0.102775)
$\hat{K}_{boot}$	17.489658	0.0010247	(17.469574, 17.509742)
$\hat{r}_{boot}$	0.702098	0.002938	(0.696339, 0.707857)
$(\hat{x_0})_{boot}$	0.0990520	0.002152	(0.094834, 0.103270)

Table 3

Asymtotic and bootstrap OLS estimates for CV data, M=0.1.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{asy}$	17.528701	0.019091	(17.491283, 17.566120)
$\hat{r}_{asy}$	0.699335	0.005201	(0.689140, 0.709529)
$(\hat{x_0})_{asy}$	0.100650	0.003851	(0.093103, 0.108198)
$\hat{K}_{boot}$	17.523374	0.019155	(17.485829, 17.560918)
$\hat{r}_{boot}$	0.699803	0.005092	(0.689824, 0.709783)
$(\hat{x_0})_{boot}$	0.100317	0.003800	(0.092869 0.107764)

Table 4

Computation times (s) for asymptotic theory vs. bootstrapping

Noise level (%)	Asymptotic theory	Bootstrapping
1	0.017320	4.285640
5	0.009386	4.625428
10	0.008806	4.914146

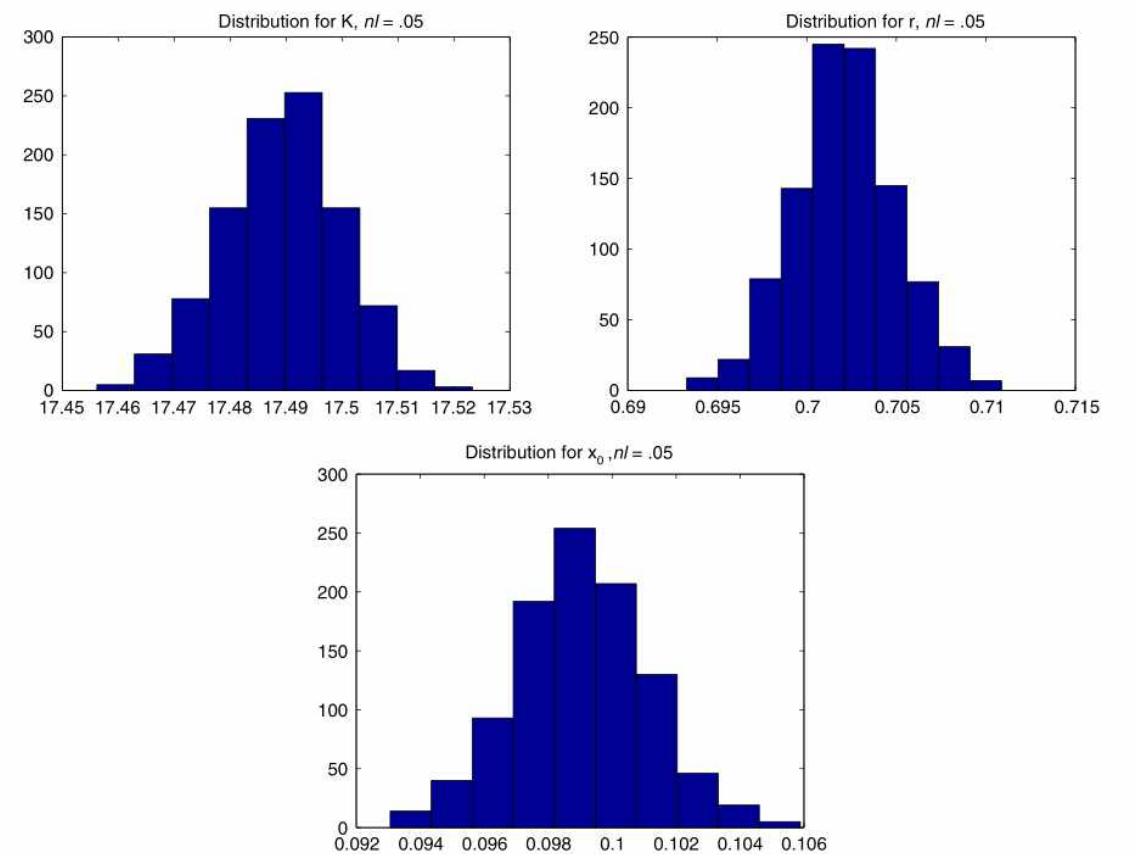


Fig. 2. Bootstrap parameter distributions corresponding to 5% noise with CV.

3-3) GLS 방법을 이용한 등분산이 아닌 경우의 데이터

GLS 공식을 이용하여 등분산이 아닌 경우의 데이터 셋에서 계산을 이끌어 낼 수 있다 (이 경우의 계산에서 우리는 GLS 방법의 반복수를 1번으로 하였다). 부트스트랩 추정량은 마찬가지로 $M=1000$으로 설정하였다. 표준오차와 이와 관련된 신뢰구간은 Table 5-7에 제시하였다. Fig. 3에서는 3-2)와 마찬가지로, 우리는 $Nl=0.05$인 경우의 계산된 값에 대한 실제적인 분포를 제시하였다. 나머지 Nl의 경우는 이와 매우 유사하므로 생략한다.

부트스트랩 방법으로부터 계산된 표준오차는 점근적 이론을 이용한 표준오차 계산 결과와 매우 유사하다는 것을 관찰할 수 있었다. 이들 각각의 경우에서, 모수 K의 표준오차 계산은 r, x_0의 표준오차보다 2~3배 더 크다. Table 8의 계산 속도에서는 여전히 부트스트랩 방법이 더 느린 속도를 가지는 것을 확인할 수 있고, 그러므로 등분산이 아닌 경우 GLS 방법에 있어 점근적 이론을 선택하는 것이 더 나은 방법이 될 것이다.

Table 5

Asymtotic and bootstrap GLS estimates for NCV data, Nl=0.01.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{asy}$	17.514706	0.028334	(17.459171, 17.570240)
$\hat{r}_{asy}$	0.70220	0.001156	(0.699934, 0.704465)
$(\widehat{x_0})_{asy}$	0.099145	0.000435	(0.098292, 0.099999)
$\hat{K}_{boot}$	17.515773	0.027923	(17.461045, 17.570502)
$\hat{r}_{boot}$	0.702136	0.001110	(0.699960, 0.704311)
$(\widehat{x_0})_{boot}$	0.099160	0.000416	(0.098344, 0.099976)

Table 6

Asymtotic and bootstrap GLS estimates for NCV data, M=0.05.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{asy}$	17.322554	0.148891	(17.030728, 17.614380)
$\hat{r}_{asy}$	0.699744	0.006126	(0.687736, 0.711752)
$(\hat{x_0})_{asy}$	0.099256	0.002313	(0.094723, 0.103790)
$\hat{K}_{boot}$	17.329282	0.146030	(17.043064, 17.615500)
$\hat{r}_{boot}$	0.700060	0.006003	(0.688294, 0.711825)
$(\hat{x_0})_{boot}$	0.099210	0.002329	(0.094645, 0.103775)

Table 7

Asymtotic and bootstrap GLS estimates for NCV data, M=0.1.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{asy}$	17.233751	0.294422	(16.656683, 17.810818)
$\hat{r}_{asy}$	0.676748	0.011875	(0.653473, 0.700024)
$(\hat{x_0})_{asy}$	0.109710	0.005015	(0.099880, 0.119540)
$\hat{K}_{boot}$	17.241977	0.275328	(16.702335, 17.781619)
$\hat{r}_{boot}$	0.676694	0.011845	(0.653479, 0.699909)
$(\hat{x_0})_{boot}$	0.109960	0.005031	(0.100098, 0.119821)

Table 8

Computation times (s) for asymptotic theory vs. bootstrapping

Noise level (%)	Asymptotic theory	Bootstrapping
1	0.030065	16.869108
5	0.032161	21.549255
10	0.037183	24.530157

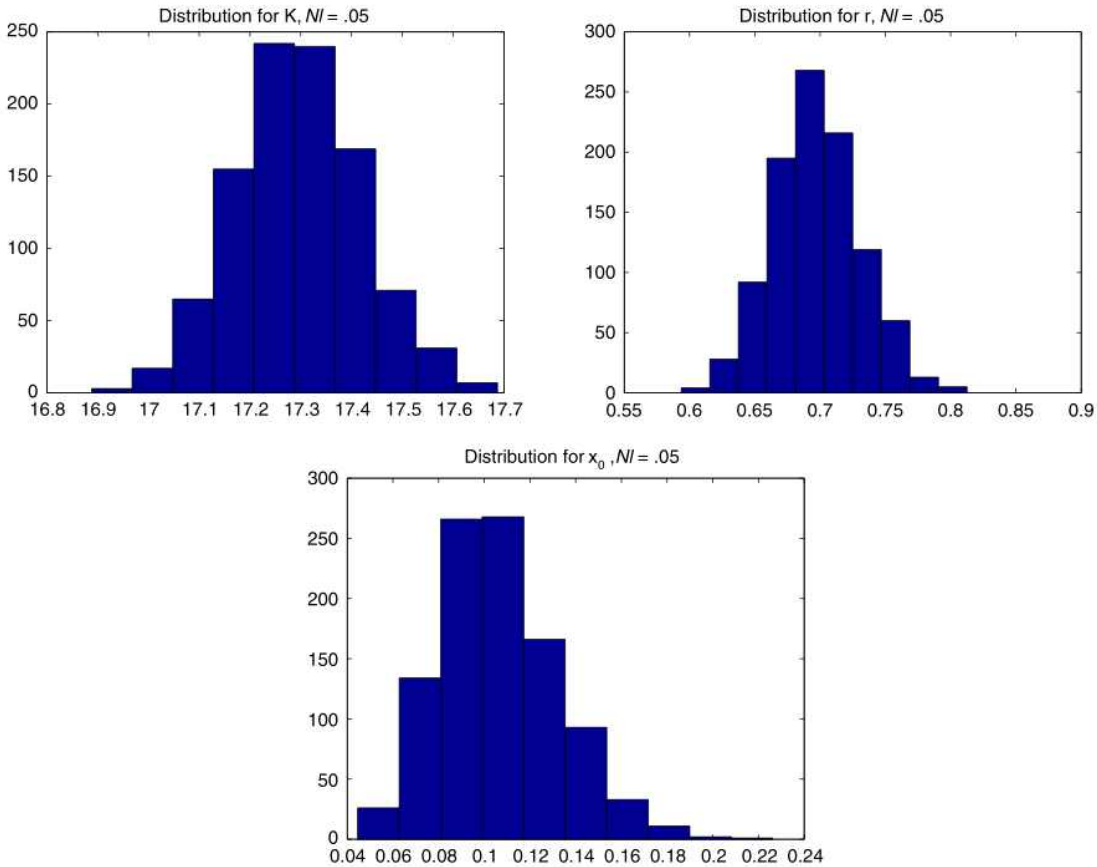


Fig. 3. Bootstrap parameter distributions for 5% noise with NCV.

4. 결론 및 제언

점근적 이론과 부트스트래핑은 모수 추정에서의 불확실성의 정량화하는 별개의 방법이다. 점근적 이론은 항상 계산적으로 부트스트래핑보다 빠르다. 그러나, 이는 두 방법 사이에 확실하게 비교되는 유일한 점이다. 데이터 셋의 유형에 따라 어느 한 방법이 다른 방법보다 더 유리할 수 있다.

OLS를 이용한 등분산 데이터의 경우, 점근적 이론 대신 부트스트래핑을 사용하는 것에 뚜렷한 이점이 없다. 그러나 복잡한 모델에 있어서는 점근적 이론에서 민감도가 필요한 계산이 매우 복잡해진다. 이 때, 부트스트래핑이 표준오차를 추정하기에 더 합리적이다. 만약 계산 속도를 이미 고려하고 있고, 민감도를 계산할 수 있다면, 점근적 이론에 이점이 있다.

주어진 통계적 모델이 등분산이 아닌 경우를 정확히 가정할 수 있을 때, 모수는 GLS 방법을 이용하여 추정되며, 점근적 이론 오차 추정치와 부트스트래핑 오차 추정치는 비교가 가능하다. 데이터의 국소적 변동이 모수 추정에 있어 중요 영역에서 크다면, 부트스트래핑 공분산 추정량이 유의미하고, 중요 영역에서 작다면 점근적 이론의 표준오차보다 표준오차의 부트스트랩 추정량이 덜 유의미한 결과를 갖게 된다.

결과적으로, 등분산이 아닌 경우의 데이터를 가정할 때, 부트스트래핑과 점근적 이론 중 무엇을 선택할 것인지는 모수의 추정을 위한 중요 영역에서 데이터의 국소적 변동에 의존한다.

등분산을 가정할 수 없는 것이 일반적인 상황이므로, GLS를 적용하는 것이 적합하고, OLS 방법은 등분산을 가정할 수 있는 경우에 적용할 수 있으므로, OLS 방법은 GLS의 특수한 경우로 간주된다. 즉, 상관관계가 존재한다면 OLS를 적용하고, 상관관계가 존재하지 않는 경우에는 GLS방법을 적용하는 것이 좀 더 합리적이다.

Ⅱ. 학습 및 분석 내용

1. 통계 이론에 대한 이해

앞선 논문 요약에서 등장하는 모델인 미분방정식과 관련한 이론에 대한 이해와, 논문을 꿰뚫고 있는 전반적인 통계 이론에 대한 이해 및 지식 습득, 그리고 논문에서 실제로 어떻게 쓰였는지 분석해보도록 한다. 통계 이론은 기초 통계학 교재와 응용수학 전공 통계학 강의에서 쓰이는 수리통계학 교재를 참고도서로 하여 학습하고, 미분방정식에 관한 내용은 현재 전공기초 과목인 미분방정식 강의에 쓰이는 교재를 참고도서로 하여 학습하도록 한다. 자세한 내용은 참고문헌에 명시하였다.

1-1) 논문 이해를 돕기 위한 기초적인 통계 이론

(1) 통계 분석

통계 분석은 연구 목적에 의해 수집된 데이터를 분석하여 결론이나 정보를 얻는 일련의 과정이다. 이 과정은 개략적으로 다음과 같이 요약된다.

*확증적(Confirmatory) 데이터 분석
① 통계적 가설(statistical hypothesis)이나 모형(model)을 설정
② 관련 데이터를 수집하여 정리 및 분석
③ 가설 혹은 모형의 유의성(significance) 검정

*탐색적 데이터 분석
수집된 데이터로부터 정보나 결론을 얻는 통계적 분석 방법

(2) 변수와 데이터

분석의 대상이 되는 데이터는 행렬의 형태로 입력되고, 열은 변수(variable), 행은 개체(subject)로 구성된다. 각 원소(셀)을 관측치(observation)라 한다.

(3) 변수의 종류

*수리 통계
-이산형(discrete) : 측정 결과를 셀 수 있는 경우 (성별, 직업, 교통량, 나이 등)
-연속형(continuous) : 측정 결과가 무한히(infinite) 많은 경우, (키, 몸무게 등)

*자료 분석
-측정형 변수(metric, measurable, quantitative) : 실험 개체의 측정 가능한 특성을 측정한 변수 (키, 몸무게, 평점, 교통량, 사망자 수 등). 연속형 변수는 모두 측정형 변수, 이산형 변수 중에는 측정형 변수가 있을 수 있다.
-분류형(범주형) 변수(categorical, classified, qualitative) : 개체를 분류하기 위해 측정된 변수 (성별, 결혼여부 등). 분류형은 다음 두 가지로 분류된다.
명목형(nominal) : 개체를 분류만 한다. (성별, 결혼여부, 학력)
순서형(ordinal) : 순서를 가진다.(성적(ABC), 소득수준(상중하), 리커트 척도(5점))

*시간
데이터가 시간적 순서를 가지면 이를 시계열 자료(time series)라고 하고, 그렇지 않은 경우 횡단면 자료(cross-section:일정 시간에 한꺼번에 조사)라 한다.

*인과 관계
설명변수(독립변수) : 통계 모형의 인과관계에서 원인이 되는 변수
반응변수(종속변수) : 통계 모형의 인과관계에서 결과나, 영향을 받는 변수

(4) 분석 방법

*일변량 분석
일변량 분석에서 관심의 대상은 모수(parameter θ)에 있으므로, $(x_1,\cdots,x_n)$으로부터 숫자 요약인 통계량(표본평균, 표본분산 등)을 구하고 그것을 이용하여 모수를 추정하거나 모수에 대한 가설(통계적 가설)을 검정하게 된다. 모수 추정에 사용되는 통계량을 추정량(estimator)라 하고, 가설검정에 사용되는 통계량을 검정통계량(test statistic)이라 한다.

모수를 추정하는 방법은 점추정과 구간추정이 있다. 구간추정과 가설 검정을 위해서는 추정치와 검정통계량의 분포에 대한 정보가 필요한데, 이는 분포를 알아야 확률을 구할 수 있기 때문이다(95% 신뢰구간, 5% 유의수준, 유의확률). 모집단의 분포를 모르고 소표본인 경우 검정통계량의 분포를 알 수 없으므로, 비모수적 방법(nonparametric method)을 사용하게 된다.

*다변량 분석
다변량 분석은 세 개 이상(이변량 분석은 2개의 관계를 가짐)의
(1)인과관계를 규명, 분석하거나(회귀분석) 분산분석(Multivariate, ANOVA),
(2)상관관계(correlation)를 이용하여 변수 축약(reduction)이나 개체 분류(classification)에 관련된 분석 방법이다.

일반적으로 협의의 다변량 분석이란, 변수의 상관관계를 이용하여 대용량 데이터의 변수를 축약하거나 개체들을 몇 개의 그룹으로 나누는데 사용되는 분석 방법을 일컫는다. 일반적으로 다변량 분석은 변수를 축약(주성분분석)하거나 그룹화(요인분석)하는 변수유도기법과 개체의 집단을 판별(판별분석)하거나 분류하는(군집분석, 다차원척도법, 대응분석) 개체유도기법으로 나뉜다.

(5) 회귀분석(regression analysis)

회귀분석은 이론이나 경험적 근거에 의해 설정된 변수들간의 (선형) 함수관계가 유의한지 알아보는 통계분석 방법이다. 회귀분석은 다음 물음에 답할 수 있다.

① 변수들 간의 함수관계가 존재하는가?
② 설정된 설명변수의 영향(설명)이 유의(significant)한가?
  - 유의한 설명변수 중 종속변수의 영향력이 가장 큰 것은?(표준화 회귀계수)
③ 최종 회귀모형을 이용하여 주어진 설명변수에 있어서 종속변수 예측치 얻음.

[회귀분석 과정]

① 종속변수의 무엇을 설명할 것인가?
  - 종속변수가 가진 정보 : 변동(분산)
② 어떤 설명변수(독립변수)가 종속변수에 영향을 미치는가?
  - 변수의 선택은 이론이나 경험에 의해 분석자가 선택
  - 영향을 미치는 모든 설명변수를 고려하는 것은 불가능하므로 오차항 존재.
③ 설명변수와 종속변수는 어떤 함수 관계를 갖는가?
④ 데이터로부터 선형 모형의 회귀계수를 추정한다.
  - 종속변수 변동=모형이 설명하는 변동 + 모형이 설명 못하는 변동
  - 수집한 데이터로 변수들 간의 함수 관계를 보기위한 산점도를 그린다.
  - 데이터에 가장 적합한 직선(곡선접합)은 어떻게 그릴 것인가? : OLS방법등
⑤ 회귀모형의 유의성 검정
  <1> 회귀모형에 고려한 설명변수 중 적어도 하나는 유의한가? : 분산분석
  <2> 개별 설명변수에 대한 유의성 : t-검정

[회귀 모형의 가정(assumption)]

① 회귀계수 α, β는 모수이며 상수(constant)이다.
② 종속변수와 설명변수 간에는 함수관계가 존재한다.
③ 설명변수 X는 확률변수가 아닌 수학변수로 오차없이 측정할 수 있다고 가정
- 회귀모형에서 확률변수는 e와 Y이다. 확률변수이면 확률분포함수를 가진다.
④ $e_i \sim i.i.d N(0, \sigma^2)$: independently and identically distributed

독립성(independent) : 오차항은 서로 독립이다. 즉 각 오차는 서로 영향을 주지 않는다. 독립성 가정은 시계열 데이터의 경우에만 체크한다.
정규성(normality) : 오차항은 정규분포를 따른다. 이 가정은 F-검정 방법(분산분석)을 사용하기 위하여 반드시 필요하다.
등분산성(homoscadicity) : 오차항의 분산은 동일하다. 분산이 일정하다는 가정의 주어진 설명변수 값에서 관측되는 Y의 값의 분산이 일정하다는 의미와 같다. 분산이 다르면 설정된 회귀모형이 적절함에도 불구하고 관측치가 직선에 모여있지 않게 된다. 분산이 크므로 벗어나는 경향이 있다.

*회귀계수의 추정
회귀 모형을 추정한다는 것은 수집된 데이터(산점도)에 가장 적절한 회귀 직선을 구하는 것이다. 방법은 OLS(Ordinary Least Square, 최소자승법)방법과 MLE(Maximum Likelihood Estimator, 최대 우도 추정법, 최우 추정법) 방법이 있다.

*최소자승법(OLS)
각 관측치에 가장 적합한 회귀 직선은 회귀 직선과 관측치의 벗어난 정도(오차)가 가장 적은 직선일 것이다. 그런데 $\sum_{i=1}^{n} e_i = 0$ 이므로, $\sum_{i=1}^{n} e_i^2$ (절대값 대신 제곱을 하는 이유 : (1) 다루기 쉽다, (2) 멀리 떨어질수록 더 큰 페널티를 부여)을 최소화하는 α, β를 추정하는 방법을 최소자승법(OLS) 라고 한다.

여기서, $Q=\sum_{i=1}^{n} e_i^2$ 을 최소화하는 추정치 $\hat{\alpha}, \hat{\beta}$를 OLS 추정치라 한다.

이 추정치들을 구하는 방법은 다음과 같이 요약할 수 있다.

Q를 α, β에 대해 각각 편미분하고, 그 결과를 0이라 놓고 얻은 연립방정식을 풀면 된다. 이를 정규방정식(normal equation)이라 한다.

$$\frac{\partial Q}{\partial \alpha} = -2\sum_{i=1}^{n}(y_i - \hat{\alpha} - \hat{\beta}x_i) = 0$$
$$\frac{\partial Q}{\partial \beta} = -2\sum_{i=1}^{n} x_i(y_i - \hat{\alpha} - \hat{\beta}x_i) = 0$$

위와 같은 정규 방정식에서 α, β의 해(OLS추정치)를 구하면 다음과 같다.

$$\hat{\beta} = \frac{\Sigma(x_i - \bar{x})(y_i - \bar{y})}{\Sigma(x_i - \bar{x})^2} \quad , \quad \hat{\alpha} = \bar{y} - \hat{\beta}\bar{x}$$

$S_{xy} = \Sigma(x_i - \bar{x})(y_i - \bar{y}), S_{xx} = \Sigma(x_i - \bar{x})^2$라 정의하면 $\hat{\beta} = \frac{S_{xy}}{S_{xx}}$ 이다.

*최대우도 추정량(MLE)

최대우도 추정량이란 우도함수(Likelihood function, 확률밀도함수의 곱)를 최대화하는 추정량이다. 확률표본 $(x_1, \cdots, x_n) \sim f(x;\theta)$ 인 경우 우도함수는 $L(\theta; x_1, \cdots, x_n) = \prod f(x_i;\theta)$이고, 이 함수를 최대화하는 $\hat{\theta}$를 최대우도 추정량(MLE)이라 한다. 회귀모형의 가정으로부터, $(y_1, \cdots, y_n) \sim N(\alpha + \beta x_i, \sigma^2)$임을 알았다. 그러므로 우도함수는

$$\begin{aligned} L(\alpha, \beta; x_1, \cdots, x_n) &= f(y_1, \cdots, y_n; \alpha, \beta) \\ &= \prod \frac{1}{\sqrt{2\pi\sigma}} exp\left(-\frac{(y_i - \alpha - \beta x_i)^2}{2\sigma^2}\right) \\ &= \left(\frac{1}{\sqrt{2\pi\sigma}}\right)^n \exp\left(-\frac{\Sigma(y_i - \alpha - \beta x_i)^2}{2\sigma^2}\right) \end{aligned}$$

우도함수를 최대화하는 α, β를 구하면 이것이 MLE 추정치이다. 우도함수를 최대

화 한다는 것은 우도함수의 로그 $\ln L \propto \Sigma(y_i - \alpha - \beta x_i)^2$를 최대화 하는 것과 동일하다. 그러므로 회귀계수에 대한 MLE추정치는 OLS추정치와 동일하다.

(6) 잔차분석(residual analysis)

*잔차(residual) : 오차항에 대한 추정치

*오차에 대한 가정(선형성, 등분산성, 정규성) 파괴와 해결책 : <등분산성>
-등분산성의 진단 : 잔차와 예측치 산점도, 나팔모양
-해결방법 : 1. 가중최소자승법(WLS, GLS) 사용,
2. 종속변수변환 (일반적으로Log변환)

*WLS(Weighted Least Square) 방법
$\min_{\alpha,\beta} \Sigma w_i (y_i - \alpha - \beta x_i)^2$ 인 $\hat{\alpha}, \hat{\beta}$를 WLS 추정치라 한다. 일반적으로 가중치 w_i는 $1/\sigma_i^2$ (σ_i^2를 알고 있을 때, 그러나 실제 알지 못한다.) 혹은 $1/x_i^2, 1/y_i^2$ 등을 사용한다. 다중회귀에서는 문제가 되는 설명변수를 이용한 가중치 $1/x_i^2$를 사용하기도 하지만 판단이 쉽지 않아서 다중회귀모형에서도 $1/\hat{y}_i^2$를 사용한다.
가중치를 $1/x_i^2$로 사용하는 경우는 회귀모형의 OLS 추정치를 구하는 문제와 동일하다. 가중치를 $1/\hat{y}_i^2$로 사용하는 경우는 정규방정식에 의해 추정치를 구할 수 있다.

(7) 분산 분석적 회귀분석

*불확실성(uncertainty)
회귀 모형에서 데이터에 포함된 불확실성이란 적합 회귀선(추정 회귀식)으로부터 관측치가 얼마나 벗어나 있는지를 의미하며, 이것에 대한 측정은 오차변동(SSE : Error Sum of Squares) 혹은 오차제곱합이라 하며, 적합 회귀식에 의해 설명되지 않는 변동에 해당된다. 이 오차 변동을 $(n-p-1)$으로 나눈 값을 MSE(Mean Squared Error)라 하면, 이는 오차의 분산에 대한 추정치로 사용한다.

1-2) 역문제(inverse problem)

방정식을 푼다는 것은 방정식의 해를 구하는 것이다. 이것은 미분방정식의 경우에도 마찬가지다. 미분방정식을 연구하는 사람의 주된 관심사는 해를 구하거나 해의 성질을 이해하는데 있다.

하지만 역문제는 해에 대한 매우 제한적인 정보를 갖고 그 방정식의 계수를 찾아내는 것이다. 즉 미분방정식의 함수해에 대한 제한된 정보로부터 그 미분방정식의 계수를 구하는 것이다. 역문제라는 이름이 붙은 것도 이러한 연유에서다. 방정식의 해를 구하지 않고 거꾸로 계수를 구하기 때문이다. 어떤 과정에 입력을 가하면 그 과정을 통한 후 출력이 나오게 된다. 보통의 문제는 어떤 출력이 나타나는지에 관심이 있다. 하지만 역문제의 관심은 입력과 출력의 관계를 보고 그 과정이 어떻게 구성돼 있는지를 파악하는데 있다.

역문제의 해를 구하는 일은 매우 어려운 일이며, 그 까닭은 역문제의 비선형성에 있다. 어떤 계가 비선형이라는 것은 입력량의 변화에 비례해서 출력량이 변하지 않는다는 것을 뜻한다. 가령 입력량을 2배로 했을 때 출력량도 2배로 늘어날 경우, 이 계는 선형이다.

우리의 논문에서 주어진 모델은 비선형 동역학계에 속하는 미분방정식으로써, 주어진 미분방정식의 해는 상미분방정식의 여러 해법 중 변수분리법을 이용하여도 손 쉽게 일반해를 구할 수 있다. 다음의 미분방정식은 논문에서 주어진 미분방정식이고, 여기서 매개변수(모수)로 취하는 것은 K, r, x_0 으로, 역문제의 관점에서는, 이 매개변수(미분방정식의 계수)의 값을 추정하는 것이 목적이다.

$$\frac{dx(t)}{dt} = rx(t)\left(1 - \frac{x(t)}{K}\right) \ , \ x(0) = x_0. \ \text{(주어진 ODE)}$$

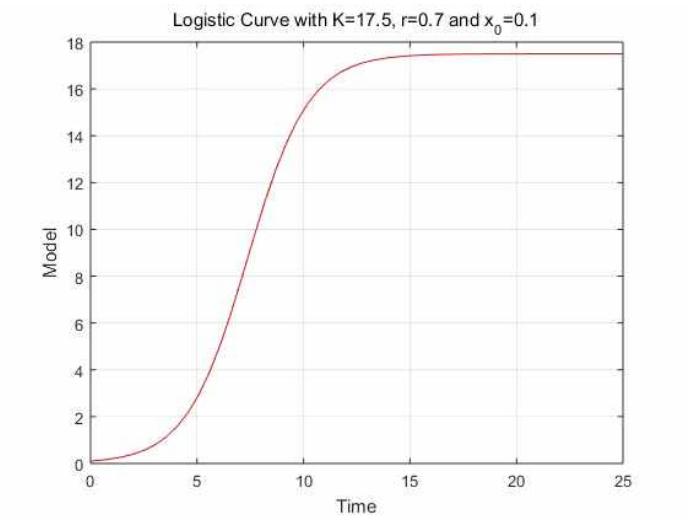
$$x(t) = f(t,\theta) = \frac{K}{1 + \left(\frac{K}{x_0} - 1\right)e^{-rt}} \ , \ \theta = (K, r, x_0) \ \text{(ODE의 일반해)}$$

이러한 역문제의 관점에서, 논문은 미리 참값을 설정하고, 그에 대한 추정 방법의 효율성과 불확실성을 정량화하는 것의 방법의 비교를 목적으로 하기 때문에 모수(매개변수)의 참값을 $\theta = (17.5, 0.7, 0.1)$로 주어 미리 설정한다. 또한 위 ODE의 일반해는 로지스틱 곡선으로써, 비선형함수이고, 주어진 참값 모수를 대입하여 참값이 될 곡선을 미리 설정한다. 그 후 두 방법(부트스트래핑 vs 점근적 이론)을 이용하여 모수 추정치를 구하고, 표준오차와 분산을 구하여 모수 추정과 불확실성 정량화에 대한 효율적인 방법이 무엇인지 분석하였다. 이 때, 부트스트래핑과 점근적 이론 모두 비선형 회귀분석에 기초하여 알고리즘을 전개하였는데, 이는 ODE의 일반해를 함수값으로 갖는 y에 대한 곡선을 일종의 반응변수로 하고, 설명변수를 t로 둔 후, 추정하는 회귀계수가 될 부분을 모수 추정치 θ로 설정하면 비선형 회귀분석 문제에서의 회귀계수를 추론하는 일이 되는 것과 같아진다.

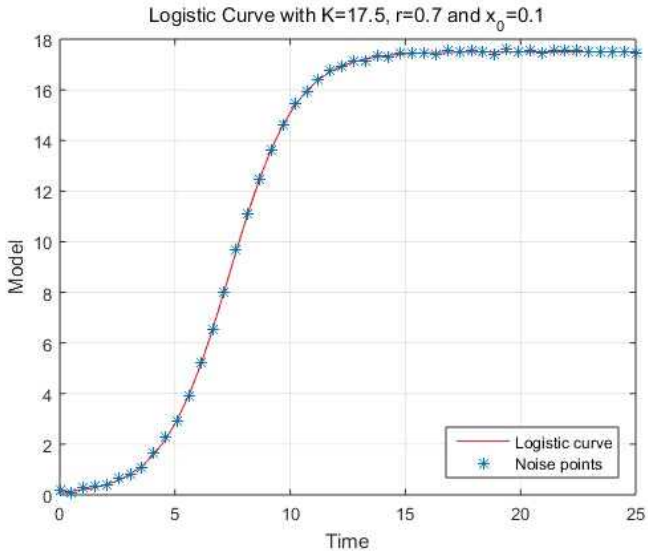
논문에서는 앞서 언급한 회귀계수(모수)의 추론을 위해 필요한 데이터 셋을 참값 곡선에 noise를 주는 방법으로 구하였는데, 이는 특정한 인구모델의 데이터가 아닌 곡선 주변으로 난수를 생성하여 만든 noise 데이터를 만들어주는 방법이다. 이 데이터는 그래프 혹은 대수적 조작의 측면에서 보면, 회귀모형에서 오차항으로 생각할 수 있고, 이 오차항은 회귀 모형에서의 가정을 따른다(정규성, 등분산성, 독립성). 그러나 이러한 오차항의 가정 중 등분산성을 만족하지 않는 경우에는 회귀계수를 구하기 어려우므로, 이를 해결하기 위한 해결책으로 GLS 방법을 제시하여 추정치 θ를 구하였다. 구한 모수 추정치를 가지고, 오차(데이터)들에 가장 최적으로 접합되는 회귀 모형의 계수가 곧 모수 추정치가 되므로, 오차의 추정치라고 생각할 수 있고, 이를 이용하여 분산 추정치를 구한 후, 오차항의 실제적인 분포를 구한다. 그러면 분포를 알고 있으므로, 신뢰구간과 표준오차를 계산할 수 있게 되고, 이를 부트스트랩 방법과 점근적 이론을 이용하여 구하여 불확실성의 정량화의 측면에서 어떤 방법에 이점이 있는지 분석하는 것이 최종적인 내용이다.

2. MATLAB을 이용한 실질적인 구현

우선 MATLAB을 이용하여 주어진 미분방정식의 일반해인 로지스틱 곡선을 그려보면 다음과 같다.



이 로지스틱 방정식에 의해 그려지는 $t=0$에서 $t=25$까지의 모수를 $(17.5, 0.7, 0.1)$을 갖는 곡선은 참값이 되는 곡선으로써, 알고리즘을 시뮬레이션 하기 위한 데이터 셋(오차항)의 기준이 되는 모형이다. 사용할 데이터 셋은 실제로 주어지는 데이터가 아닌 난수생성기를 이용하여 참값 곡선에 noise를 주는 방식으로 구하려고 한다. 이는 실제 회귀모형의 관측값(확률분포함수)으로써, $y_j = f(t_j, \theta_0) + \epsilon_j$인 형태를 가진다. 여기서 f는 참값 곡선이고, ϵ_j가 noise가 된다. 이 noise는 논문에서 level이 5%인 경우, 즉 오차항이 0.05인 경우의 noise에 대한 결과만 표시하였다. 이제 이러한 noise set을 구하고자 한다.



앞서 구한 로지스틱 곡선에 $\mathcal{N}=0.05$인 잡음을 주었을 때 위와 같이 곡선 주변에 noise point가 생성되는 것을 확인할 수 있다. 이를 주어진 데이터라 가정하고 앞으로의 이론들을 전개해 나갈 것이다. 목차에 표시하지 않았으나, 두 방법을 적용한 결과를 합리적으로 확인할 수 있도록, 소제목과 같은 부분을 굵은 글씨로 표시할 것이다.

(1) 등분산을 가정한 경우(CV)의 OLS 부트스트래핑 알고리즘

MATLAB 프로그래밍을 통하여 실질적인 부트스트래핑 알고리즘을 실행한 결과값은, 논문의 Table 양식과 마찬가지로, 다음과 같이 구하였다.

Table 1

Bootstrap OLS estimates for CV data, $\mathcal{N}$=0.01.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{boot}$	17.516934	0.018183	(17.481295, 17.552573)
$\hat{r}_{boot}$	0.700754	0.003763	(0.693378, 0.708130)
$(\widehat{x_0})_{boot}$	0.100052	0.002568	(0.095018, 0.105086)

Table 2

Bootstrap OLS estimates for CV data, M=0.05.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{boot}$	17.519803	0.021869	(17.476938, 17.562668)
$\hat{r}_{boot}$	0.700469	0.007163	(0.686428, 0.714510)
$(\hat{x_0})_{boot}$	0.100300	0.006172	(0.088203, 0.112398)

Table 3

Bootstrap OLS estimates for CV data, M=0.1.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{boot}$	17.504109	0.030023	(17.445263, 17.562954)
$\hat{r}_{boot}$	0.701875	0.012291	(0.677842, 0.725966)
$(\hat{x_0})_{boot}$	0.099066	0.010489	(0.078506, 0.119626)

MATLAB으로 구현한 부트스트래핑 알고리즘으로 각 잡음 레벨별로 구한 결과값을 관찰해보자. 고정된 참값 모수인 $\theta = (17.5, 0.7, 0.1)$에 대하여, 추정치 $\hat{\theta}$ 값은 대부분 소수점 이하 1자리까지 정확하다. 또한, 이러한 추정치 $\hat{\theta}$를 이용하여 구한 표준오차와 95% 신뢰구간도 모수 추정 위치를 잘 설명하고 있음을 보여준다.

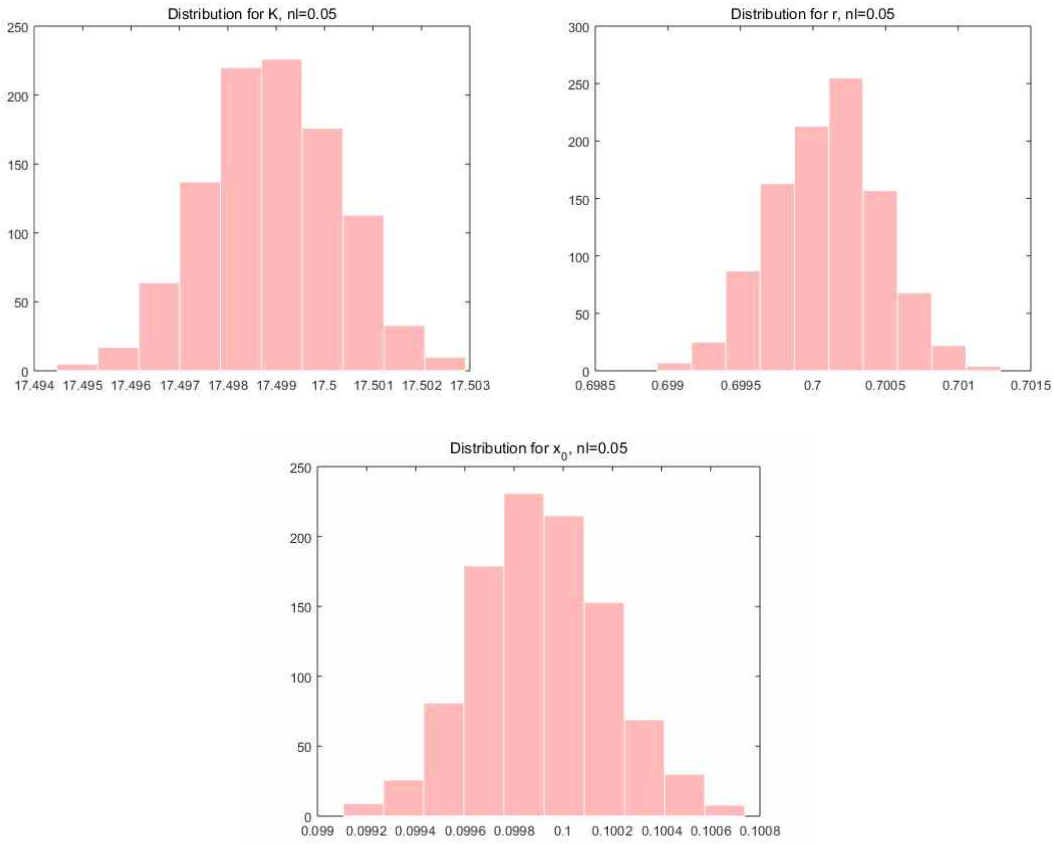
계산속도는 매트랩 함수인 tic/toc을 이용하여 측정하였으며, 실제 논문에서의 부트스트래핑 계산속도가 약 4초대인 것을 감안하면, 학습 결과의 부트스트래핑 계산속도와는 큰 차이가 있다. 논문과 마찬가지로 실제로 구한 추정치들이 잘 근사하고 있으므로, 이는 코드의 효율성이 떨어지거나, 최소가 되게 하는 점을 찾는 함수의 선택, x축 변수인 시간 벡터의 길이에 따라 달라지게 된다. 실제로 구한 각 노이즈 레벨에 대한 부트스트래핑 계산 속도는 아래와 같이 구하였다.

Table 4

Computation times (s) for bootstrapping in OLS

Noise level (%)	Bootstrapping
1	7.746404
5	7.785673
10	9.815415

$Nl = 0.05$인 경우의 부트스트래핑 알고리즘을 이용하여 얻은 모수 추정치에 관한 분포는 다음과 같다.



모수 추정치 $\hat{\theta} = (\hat{K}, \hat{r}, \hat{x_0})$의 각 성분에 관한 분포를 관찰하면, 분포가 정규성을 띄고, 평균이 모수의 참값에 근접하는 양상을 보이는 것을 확인할 수 있다.

(2) 등분산이 아닌 경우(NCV)의 GLS 부트스트래핑 알고리즘

등분산이 아닌 경우에도 마찬가지로 구할 수 있다. 단, GLS방법을 사용하고, 회귀모형에 대한 전제 가정을 조금 수정하여 이용한다. 특히 GLS방법은 가중치를 주는 가중최소제곱법이므로, 가중치에 대한 선택도 포함된다. 논문에서는 가중치를 $1/y^2$ 꼴인 반응변수에 대한 가중치를 주었다. 그러면 이는 정규방정식을 통한 추정치를 구하게 되므로, 다음과 같은 정규방정식을 θ_{GLS}에 관하여 풀어 얻어진다.

$$\sum_{j=1}^{n} w_j [\, Y_j - f(t_j, \theta_{GLS})] \bigtriangledown f(t_j, \hat{\theta}) = 0 \ , \ \text{where} \ \ w_j = f^{-2}(t_j, \theta_{GLS}).$$

위 정규방정식을 유도하기 위한 기본 가정들과 수식은 논문요약 page에서 참조할 수 있다. 이제 정규방정식을 풀어 얻은 식을 잘 정리하면, 원하는 추정량인 $\hat{\theta}_{GLS}$를 얻을 수 있음을 논문에서 서술하였다. 등분산성을 만족하지 않을 때, 산점도와 회귀모형을 생각해보면, 오차항들이 등분산이 아니므로 회귀모형 주변에 불규칙적으로 퍼져있을 것이고, 이는 곧 모형의 접합이 제대로 이루어지지 않음을 고려할 수 있다. 따라서, 등분산이 아닌 경우에는 OLS방법을 이용하는 것은 적절하지 않고, 모형과 거리가 더 먼 오차항에 대한 가중치를 두어 구하는 GLS 방법이 적절함은 쉽게 생각할 수 있다.
이러한 배경에서, 직접 MATLAB 프로그래밍을 통하여 실질적으로 구현한 결과값들을 아래와 같이 제시하였다. 표준오차, 신뢰구간은 CV인 경우와 같은 공식으로 구할 수 있다.

Table 1

Bootstrap GLS estimates for NCV data, M=0.01.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{boot}$	17.527813	0.233004	(17.071125, 17.984501)
$\hat{r}_{boot}$	0.706952	0.115032	(0.481488, 0.932415)
$(\widehat{x_0})_{boot}$	0.103641	0.098708	(-0.089826, 0.297109)

Table 2

Bootstrap GLS estimates for NCV data, $\mathcal{N}$=0.05.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{boot}$	17.527055	0.241803	(17.053121, 18.000990)
$\hat{r}_{boot}$	0.705166	0.118306	(0.473286, 0.937047)
$(\widehat{x_0})_{boot}$	0.105288	0.102703	(-0.096009, 0.306586)

Table 3

Bootstrap GLS estimates for NCV data, $\mathcal{N}$=0.1.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{boot}$	17.531845	0.265714	(17.011045, 18.052646)
$\hat{r}_{boot}$	0.700993	0.130958	(0.444315, 0.957671)
$(\widehat{x_0})_{boot}$	0.110671	0.114475	(-0.113701, 0.335044)

GLS방법으로 구한 추정치 값을 잘 살펴보면, 모수의 추정치는 참값에 근사하는 경향을 보여주었으나, 표준오차를 관찰해보면 OLS방법보다 훨씬 큰 오차가 발생하고, 이에 따라 95% 신뢰구간의 폭이 매우 넓어지게 된다. 이는 Code 상의 문제도 있으나 실질적으로 훨씬 효율적으로 오차를 줄이는 방법은 결국 시간에 대한 벡터의 크기에 달려있다. OLS in CV와 GLS in NCV인 경우 모두, 벡터 t의 크기를 $n=50$으로 두고 수행하였는데, 만약 $n=1000$으로 둔 경우에 표준오차의 실제 계산값이 위 결과들보다 훨씬 작아지게 됨을 확인할 수 있다. 그러한 결과는 Table 5에 비교하여 기술하였다. 아래는 GLS방법인 경우의 계산시간을 표시한 결과로, 논문과는 달리 노이즈 레벨에 따라 계산 시간이 증가하지 않았다.

Table 4

Computation times (s) for bootstrapping in GLS

Noise level (%)	Bootstrapping
1	11.910531
5	12.107679
10	11.939790

Table 5

Bootstrap GLS estimates for NCV data, $\mathcal{M}$=0.05, ① $n=50$ vs ② $n=1000$

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
① $\hat{K}_{boot}$	17.527055	0.241803	(17.053121, 18.000990)
① $\hat{r}_{boot}$	0.705166	0.118306	(0.473286, 0.937047)
① $(\hat{x_0})_{boot}$	0.105288	0.102703	(-0.096009, 0.306586)
② $\hat{K}_{boot}$	17.514899	0.057610	(17.401983, 17.627816)
② $\hat{r}_{boot}$	0.701388	0.027306	(0.647868, 0.754909)
② $(\hat{x_0})_{boot}$	0.100072	0.023268	(0.054465, 0.145679)

Table 5를 관찰하면, 표본이 크면 클수록 표준오차가 이전의 결과보다 상당히 줄어들었음을 확인할 수 있다. 표준오차가 작아진 관계로, 95%신뢰구간 또한 구간의 폭이 확연히 줄어들었음을 다음 그래프에서 확인하였다.

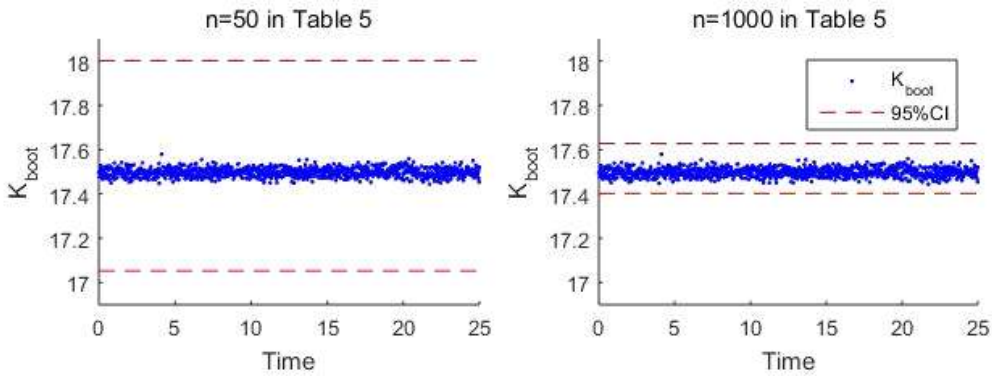
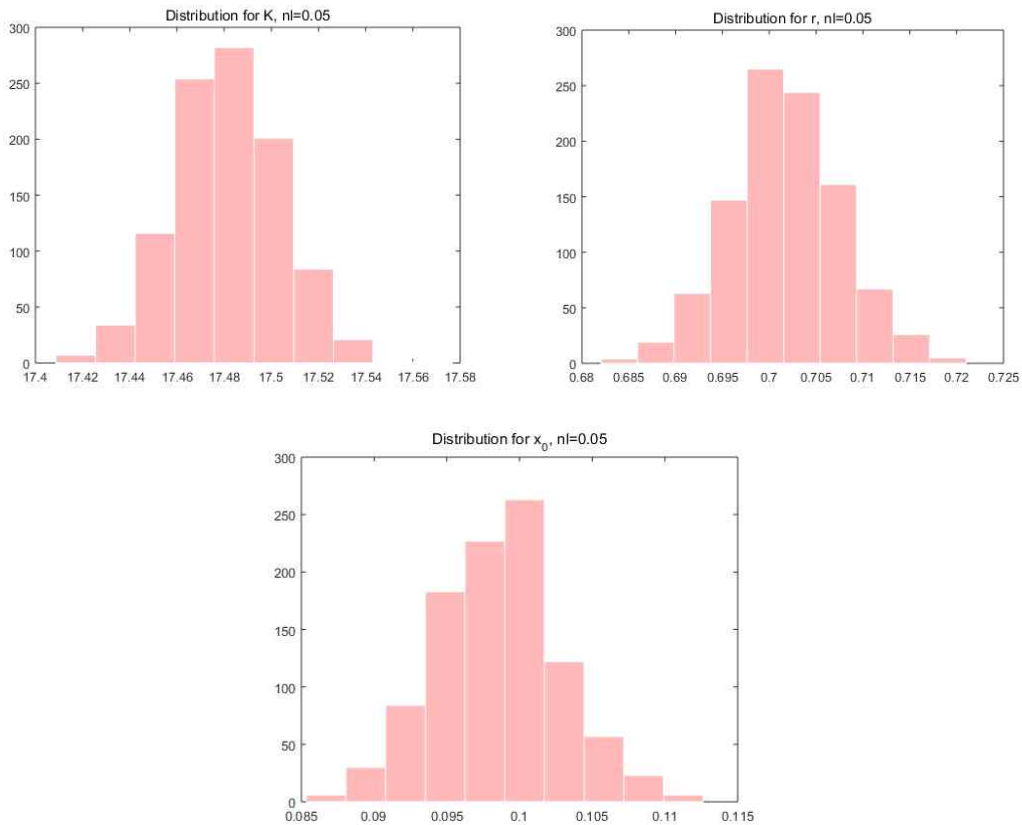


Table 5의 내용 중 $\hat{K}_{boot}$에 대한 95%신뢰구간을 위와 같이 비교하였다. $\hat{K}_{boot}$는 결국 n이 커질수록 모수에 근사하게 되므로, $n=50$인 경우의 $\hat{K}_{boot}$를 그대로 이용하였고, 실제적인 신뢰구간만 시각적으로 비교하여도 무방하므로, 각 경우의 95%신뢰구간만 출력하였다. 추정치 $\hat{K}_{boot}$를 생각하지 않고 신뢰구간을 살펴보면 $n=1000$일 때 신뢰구간이 얼마나 근접한 구간으로 좁혀지는지 쉽게 확인할 수 있다.

$Nl = 0.05$인 경우의 부트스트래핑 알고리즘을 이용하여 얻은 모수 추정치에 관한 분포는 다음과 같다.



등분산이 아님을 가정한 경우의 GLS방법을 이용한 모수 추정치 $\hat{\theta} = (\hat{K}, \hat{r}, \widehat{x_0})$의 각 성분에 관한 분포를 관찰하면, 분포가 정규성을 띄고, 평균이 모수의 참값에 근접하는 양상을 보이는 것을 확인할 수 있다.

(3) 등분산을 가정한 경우(CV)의 점근적 이론

점근적 이론은 주어진 미분방정식(또는 적분방정식 등 역문제에서 제시되는 모델)의 해석적 해를 구하기 어려울 때 사용하는 근사적인 추정방법으로써, 모델 자체가 복잡한 경우에는 비효율적인 방법이다. 이는 주어진 미분방정식에 대한 다음과 같은 민감도 방정식을 이용하여 민감도 행렬을 구한 후, 이에 대한 표준오차 계산을 한다.

$$\frac{d\chi(t)}{dt} = F(\chi(t,\theta),\theta) = r\chi(t)\left(1-\frac{\chi(t)}{K}\right) \text{ with } \frac{d}{dt}\frac{\partial\chi}{\partial\theta} = \frac{\partial F}{\partial\chi}\frac{\partial\chi}{\partial\theta} + \frac{\partial F}{\partial\theta} \text{ (sensitivity eq.)}$$

이 때, 표준오차 계산을 위한 첫 번째 추정치 $\hat{\theta}$는 OLS / GLS 방법을 통해 구한다. 위와 같은 민감도 방정식을 이용하기 위해서는, 우선 주어진 미분방정식의 수치적인 해를 구한 후, 이에 대한 잔차를 구하여야 한다. 이는 주어진 비선형 회귀 모델에 오차항(noise)를 준 식에 초기 추정치 θ_0를 대입하여 구하고, 임의의 초기 추정치 θ를 대입하여 구한 비선형 회귀식과의 잔차를 구하면 된다. 점근적 이론에 대한 실질적인 결과는 부트스트래핑 추정 결과와 비교하여 제시할 것이다.

Table 1

Asymtotic and bootstrap OLS estimates for CV data, M=0.01.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{asy}$	17.495087	0.019199	(17.457456, 17.532719)
$\hat{r}_{asy}$	0.701105	0.001532	(0.698102, 0.704108)
$(\hat{x_0})_{asy}$	0.099235	0.001532	(0.096232, 0.102238)
$\hat{K}_{boot}$	17.516934	0.018183	(17.481295, 17.552573)
$\hat{r}_{boot}$	0.700754	0.003763	(0.693378, 0.708130)
$(\hat{x_0})_{boot}$	0.100052	0.002568	(0.095018, 0.105086)

Table 2

Asymtotic and bootstrap OLS estimates for CV data, Nl=0.05.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{asy}$	17.511290	0.072708	(17.368781, 17.653798)
$\hat{r}_{asy}$	0.702660	0.005793	(0.691304, 0.714016)
$(\hat{x_0})_{asy}$	0.097538	0.005793	(0.086182, 0.108895)
$\hat{K}_{boot}$	17.519803	0.021869	(17.476938, 17.562668)
$\hat{r}_{boot}$	0.700469	0.007163	(0.686428, 0.714510)
$(\hat{x_0})_{boot}$	0.100300	0.006172	(0.088203, 0.112398)

Table 3

Asymtotic and bootstrap OLS estimates for CV data, Nl=0.1.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{asy}$	17.520139	0.121131	(17.282721, 17.757571)
$\hat{r}_{asy}$	0.693259	0.009710	(0.674227, 0.712290)
$(\hat{x_0})_{asy}$	0.103897	0.009710	(0.084866, 0.122929)
$\hat{K}_{boot}$	17.504109	0.030023	(17.445263, 17.562954)
$\hat{r}_{boot}$	0.701875	0.012291	(0.677842, 0.725966)
$(\hat{x_0})_{boot}$	0.099066	0.010489	(0.078506, 0.119626)

Table 4

Computation times (s) for asymptotic theory vs. bootstrapping

Noise level (%)	Asymptotic theory	Bootstrapping
1	0.497234	7.746404
5	0.503647	7.785673
10	0.492820	9.815415

CV인 경우 점근적 이론을 이용한 결과를 잘 살펴보면, OLS방법을 통한 추정치 및 표준오차 계산이 부트스트래핑과 비교했을 때 양호한 추정값을 계산할 수 있었으며 표준오차도 잘 계산되었음을 확인할 수 있다. 또한, Table 4를 참조하면 부트스트래핑 방법의 계산속도는 점근적 이론을 사용한 방법보다 훨씬 느리다. 점근적 이론의 경우는 모형의 복잡도에 따라 계산하기 매우 어렵고, 예측값 또한 구하고자 하는 추정치와 거리가 먼 결과를 낳기도 하지만, 논문에서 제시한 모델에서는 잘 구현됨을 확인할 수 있다. 따라서 이러한 경우에 점근적 이론이 부트스트래핑보다 더 적합하다고 할 수 있다. 이러한 결과는 점근적 이론이 주어진 표본공간에서 무작위 복원 추출 후, 계속해서 추정치를 구하여 추정치의 분포를 알아내는 과정이 아닌, 미분방정식 자체에 기반하여 풀어내고 있기 때문에 모형의 복잡도에 의존하게 되는 것은 당연하다. 그러나 부트스트래핑의 경우는 해석적 해를 안다고 가정하고 추정하였기 때문에 초기 OLS / GLS 추정치를 구하는 것이 어렵지 않았다. 만약 부트스트래핑에서 해석적 해를 구하지 못한다고 해도, 다양한 방법으로 근사해를 구하여 추정할 수 있기 때문에 이는 문제가 되지 않는다. 왜냐면 앞서 설명하였듯, 점근적 이론은 결국 미분방정식과 민감도 행렬에 의존하므로 초기 추정치 θ_0을 구하는 것에 장애가 되지 않는다. 따라서 논문에서 앞서 언급한 것처럼, 점근적 이론은 방정식이 복잡하지 않을 때 사용하기 적절하다.

(4) 등분산이 아닌 경우(NCV)의 점근적 이론

등분산이 아닌 경우, 부트스트래핑 방법과 마찬가지로 GLS방법을 통하여 추정치 θ를 구한다. 사용할 비선형 회귀 모형도 부트스트래핑과 같은 가정을 하는 것에 무리가 없다. 단, GLS방법을 이용할 경우, 가중치 w_j를 정의하므로 후에 민감도 행렬을 계산할 시 가중치 행렬 $W(\hat{\theta})$를 정의하여 이를 곱해주어야 한다. 논문에서 제시한 정의는 다음과 같다.

$$\Sigma_0^n \approx \hat{\Sigma}^n(\hat{\theta}) = \sigma_{OLS}^2 [\chi^T(\hat{\theta}) W(\hat{\theta}) \chi(\hat{\theta})]^{-1} \quad , \quad W^{-1}(\theta) = diag(f^2(t_1, \theta, \cdots, f^2(t_n, \theta)).$$

이러한 가중치 행렬과 가중치에 주의하며, CV인 경우와 마찬가지로 추정치들을 구하면 된다. 여기서 가중치 행렬은 직접적으로 제시되어 있지 않지만 다음과 같이 구할 수 있다.

$$W^{-1}(\theta) \Rightarrow (W^{-1}(\theta))^{-1} = W(\theta) \text{ since } W: \text{ invertible \& diagonal matrix}$$

대각행렬인 가중치 행렬 W는 역행렬을 취했을 때 그 주대각성분의 역수를 취하면 W^{-1}를 구할 수 있다. 실제적인 구현에서는, 행렬 차원에 주의하며 계산하도록 한다. 등분산을 가정하지 않은 경우의 점근적 이론을 이용한 GLS 추정치들과 표준오차, 신뢰구간을 구한 결과는 다음과 같다.

Table 5

Asymtotic and bootstrap GLS estimates for NCV data, N=0.01.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{asy}$	17.402674	0.127282	(17.153200, 17.652148)
$\hat{r}_{asy}$	0.635662	0.008075	(0.619834, 0.651491)
$(\hat{x_0})_{asy}$	0.182595	0.008075	(0.166767, 0.198424)
$\hat{K}_{boot}$	17.527813	0.233004	(17.071125, 17.984501)
$\hat{r}_{boot}$	0.706952	0.115032	(0.481488, 0.932415)
$(\hat{x_0})_{boot}$	0.103641	0.098708	(-0.089826, 0.297109)

Table 6

Asymtotic and bootstrap GLS estimates for NCV data, Nl=0.05.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{asy}$	17.242934	0.072708	(17.368781, 17.653798)
$\hat{r}_{asy}$	0.721513	0.005793	(0.691304, 0.714016)
$(\hat{x_0})_{asy}$	0.098450	0.005793	(0.0861827, 0.108895)
$\hat{K}_{boot}$	17.527055	0.241803	(17.053121, 18.000990)
$\hat{r}_{boot}$	0.705166	0.118306	(0.473286, 0.937047)
$(\hat{x_0})_{boot}$	0.105288	0.102703	(-0.096009, 0.306586)

Table 7

Asymtotic and bootstrap GLS estimates for NCV data, Nl=0.1.

θ	$\hat{\theta}$	$SE(\hat{\theta})$	95% CI
$\hat{K}_{asy}$	17.251259	0.086815	(17.081102, 17.421417)
$\hat{r}_{asy}$	0.714596	0.005421	(0.703970, 0.725223)
$(\hat{x_0})_{asy}$	0.078398	0.005421	(0.067772, 0.089024)
$\hat{K}_{boot}$	17.531845	0.265714	(17.011045, 18.052646)
$\hat{r}_{boot}$	0.700993	0.130958	(0.444315, 0.957671)
$(\hat{x_0})_{boot}$	0.110671	0.114475	(-0.113701, 0.335044)

Table 8

Computation times (s) for asymptotic theory vs. bootstrapping

Noise level (%)	Asymptotic theory	Bootstrapping
1	0.316362	11.910531
5	0.321598	12.107679
10	0.337155	11.939790

Table 5부터 Table 8까지 실질적인 계산 결과를 살펴보면, GLS방법을 적용하였을 때 점근적 이론을 사용한 방법이 부트스트래핑 방법보다 예측한 추정치가 고르지 못함을 보여준다. 이는 비선형 회귀모형식이 바뀐 영향보다는 가중치 행렬의 존재 때문에 참값과의 오차가 커지기 때문이다. 계산속도는 앞선 결과들과 마찬가지로 점근적 이론이 부트스트래핑보다 훨씬 큰 이점을 갖는다. 또한 noise level에 따라 계산속도가 점점 느려짐을 확인할 수 있다. r과 x_0에 대한 추정치의 표준오차가 동일한 값으로 나오는 것처럼 결과가 나타났는데, 이는 소숫점 이하 6자리 이상 표시하지 않았기 때문이다. 예를 들면, noise level이 0.1인 경우, r, x_0에 대한 표준오차값은 각각 0.00542156642020, 0.00542156769907이고, 이를 소숫점 이하 6자리까지만 표시했기 때문이다. 이러한 이유는 앞선 결과에서도 마찬가지로 적용된다.

GLS방법을 적용한 NCV인 경우의 점근적 이론 방법에서 참값과 추정값에 대한 오차가 심한 것은 표본의 개수를 늘리면 해결된다. 본 보고서에서 사용한 벡터의 크기는 $n=50$인 경우이므로, $n=1000$로 표본 크기를 늘린다면 참값과 매우 근사한 추정값을 얻을 수 있다. 이는 점근적 이론에서, 표본이 클수록 정규분포에 근사하는 성질에 따른 것이다. 그러나 부트스트래핑 또한 표본 개수가 늘어날수록 그 정확도가 커짐음 고려하면, 이러한 현상은 표본 개수에 따른 점근적 이론의 장점으로 간주하기에 다소 부족함이 있다. 또한, 가중치는 미분방정식의 원함수에 의존하게 되므로, 모형이 복잡함에 따라 가중치에 대한 오차도 충분히 크게 발생할 수 있으며, 실질적으로 해석적인 해를 구할 수 없는 경우에 사용하기 때문에 수치적인 해를 구하는 동안 전파된 오차, 누적오차에 대한 고려를 해야하므로, 오차는 더 크게 발생하게 된다. 따라서 앞서 확인했던 CV인 경우의 부트스트래핑과의 비교와는 다르게, 어떠한 경우에도 이 논문에서는 부트스트래핑 방법에 비해 점근적 이론이 갖게되는 장점은 계산속도 이외에는 찾아볼 수 없다.

3. MATLAB Code

[Bootstrap_OLS_CVdata_Estimates.m]

```
%%%%% Bootstrapping using OLS for Constant Variance data %%%%%

clear all
clc

%%%%% Setting for the variables %%%%%
tic                                  %computation time start

t=linspace(0,25,50);                          % time variable
theta=[17 0.6 0.1];                  % initial value of theta
theta_0=[17.5 0.7 0.1];               % true value of theta = (K, r, x0)

p=3;                                % the dimension of parameter space
n=length(t);                         % the length of the vector t
m=1000;                             % the number of bootstrap sample

lowb=[0 0 0];                        % lower bound of parameter space
uppb=[inf inf inf];                   % upper bound of parameter space

iterset=optimset('TolX',1.0e-4,'MaxFunEvals',10000);

% optimset : 최적화 option 구조체 생성 또는 편집
%           최적화 함수 fminsearch,lsqcurvefit 과 같은 함수의 입력 인수로 전달
할 수 있는 options 구조체를 생성
%          (반복횟수 설정을 위한 인수를 생성하여 fminsearch 로 전달)
% *** TolX : 양의 스칼라값 / 현재 점 x 에 대한 종료 허용오차
% *** MaxFunEvals : 양의 정수 / 허용되는 함수 실행의 최대 횟수
```

```
%%% first row setting [hat{theta0} setting] %%%
% **부트스트래핑 표본은 n개의 observation 에서 각 하나의 표본이 m개의 표본을 갖는 하나의 모집단으로 만들기 위함.

y(1,:) = LogisticModel(theta_0,t);      % y is a value of random process Y.
                                        % LogisticModel is defined by user

Y(1,:) = y(1,:)+(0.1)*randn(1,n);
% *** What does y(1,:) mean? ***
% If you define a bidimensional array y, and you want to access all its elements on the first column: y(:,1) will do it.
% If you want to access to all the elements of the fift row:
% y(5,:) is the syntax you have to use. The colon means: take all the elements along the specified dimension

% Y : random process, epsilon = 0.05 since the noise level =0.05 in paper.
% 참값인 로지스틱 곡선에 noise 를 주어 임의의 가상 data를 생성해야하므로
% 오차항이 곧 곡선의 noise가 되고, 이러한 noise 를 i.i.d 한 무작위 추출을 해야하므로
% randn (정규분포된 난수 생성, 행 1, 열은 n : vector t의 크기) 을 곱하여준다.

residual(1,:)=sqrt(n/(n-p)).*(Y(1,:)-y(1,:)); % standardized residuals
r(1,:)=randsample(residual(1,:),n,'true');
% 잔차(residual) 값 중에서 t의 크기만큼 무작위복원추출(default : 'false')

theta1(1,:)                                                                    =
lsqcurvefit(@LogisticModel,theta,t,Y(1,:),lowb,uppb,iterset);  % theta1 = (K, r, x0)
% lsqcurvefit : Solve nonlinear curve-fitting (data-fitting) problems in least-squares sense
% x = lsqcurvefit(fun,x0,xdata,ydata,lb,ub,options)
```

```
%%% 2nd ~ 1000th row setting %%%

for i=1:m
    y(i+1,:)=LogisticModel(theta1(i,:),t);          % 2~1000 까지의 theta1
에 관한 y 생성
    r(i+1,:)=randsample(residual(1,:),n,'true');        % 2~1000 까지의 잔
차 생성
    Y(i+1,:)=y(1,:) + r(i+1,:);                     % 2~1000 까지의 random
process : Y = y + r

theta1(i+1,:)=lsqcurvefit(@LogisticModel,theta1(i,:),t,Y(i+1,:),lowb,u
ppb,iterset); % 2~1000 까지의 theta 추정값
end

%%% Bootstrap mean, standard error, CI using parameter %%%
format long
theta_boot=(1/m)*sum(theta1) % Bootstrap mean

col_1=theta1(:,1)-theta_boot(1,1);
col_2=theta1(:,2)-theta_boot(1,2);
col_3=theta1(:,3)-theta_boot(1,3);
Covmatrix=[col_1 col_2 col_3];

Covariance_boot=(1/(m-1))*sum((Covmatrix)'*(Covmatrix));   %  Bootstrap
Covariance

StandardError_boot=sqrt(diag(Covariance_boot))  %  Standard  error  for
bootstrapping

% 95% confidence intervals from bootstrapping
CI_lowerK=theta_boot(1,1)-1.96*StandardError_boot(1,1);
CI_upperK=theta_boot(1,1)+1.96*StandardError_boot(1,1);
CI_K=[CI_lowerK CI_upperK];
```

```
CI_lowerR=theta_boot(1,2)-1.96*StandardError_boot(1,2);
CI_upperR=theta_boot(1,2)+1.96*StandardError_boot(1,2);
CI_R=[CI_lowerR CI_upperR];

CI_lowerX=theta_boot(1,3)-1.96*StandardError_boot(1,3);
CI_upperX=theta_boot(1,3)+1.96*StandardError_boot(1,3);
CI_X=[CI_lowerX CI_upperX];

CI_ALL=[CI_K ; CI_R ; CI_X] % CI=[theta-1.96se theta+1.96se]

toc                         % computation time end
```

[Bootstrap_GLS_NCVdata_Estimates.m]

```
%%%%% Bootstrapping using GLS for Non-Constant Variance data %%%%%

clear all
clc

%%%%% Setting for the variables %%%%%
tic                                 %computation time start

t=linspace(0,25,50);                        % time variable
theta=[17 0.6 0.1];                 % initial value of theta
theta_0=[17.5 0.7 0.1];              % true value of theta = (K, r, x0)

p=3;                               % the dimension of parameter space
n=length(t);                        % the length of the vector t
m=1000;                            % the number of bootstrap sample

lowb=[0 0 0];                       % lower bound of parameter space
uppb=[inf inf inf];                  % upper bound of parameter space

iterset=optimset('TolX',1.0e-4,'MaxFunEvals',10000);

%%% first row setting [hat{theta0} setting] %%%

y(1,:) = LogisticModel(theta_0,t);      % y is a value of random process
Y.

Y(1,:) = y(1,:)+(1+(0.05))*randn(1,n);

Weights(1,:)=1./(y(1,:).^2); % initial value of weights
residuesum = @(x) sum(Weights(1,:).*(Y(1,:)-LogisticModel(x,t)).^2);  %
가중치 오차제곱합 hat{theta}
% theta1(1,:) = fminsearch(residuesum,theta);                          %
```

```
theta1 = (K, r, x0)
theta1(1,:)                                                              =
lsqcurvefit(@LogisticModel,theta,t,Y(1,:),lowb,uppb,iterset);
%%% 2nd ~ 1000th row setting %%%

for i=1:m
    y(i+1,:)=LogisticModel(theta1(i,:),t);          % 2~1000 까지의 theta1
에 관한 y 생성
    Y(i+1,:) = y(1,:)+(1+(0.05))*randn(1,n);        % 2~1000 까지의 잔차 생
성
    Weights(i+1,:)=1./(y(i+1,:).^2);               % 2~1000 까지의 weights
    residuesum                              =                          @(x)
sum(Weights(i+1,:).*(Y(i+1,:)-LogisticModel(x,t)).^2);  %  2~1000  까지의
오차제곱합(GLS)
 %    theta1(i+1,:) = fminsearch(residuesum,theta1(i,:)); % 2~1000 까지의
theta 추정값

theta1(i+1,:)=lsqcurvefit(@LogisticModel,theta1(i,:),t,Y(i+1,:),lowb,u
ppb,iterset);
end


%%% Bootstrap mean, standard error, CI using parameter %%%
format long
theta_boot=(1/m)*sum(theta1) % Bootstrap mean

col_1=theta1(:,1)-theta_boot(1,1);
col_2=theta1(:,2)-theta_boot(1,2);
col_3=theta1(:,3)-theta_boot(1,3);
Covmatrix=[col_1 col_2 col_3];

Covariance_boot=(1/(m-1))*sum((Covmatrix)'*(Covmatrix));   %   Bootstrap
Covariance

StandardError_boot=sqrt(diag(Covariance_boot))   %   Standard   error   for
```

```
bootstrapping

% 95% confidence intervals from bootstrapping
CI_lowerK=theta_boot(1,1)-1.96*StandardError_boot(1,1);
CI_upperK=theta_boot(1,1)+1.96*StandardError_boot(1,1);
CI_K=[CI_lowerK CI_upperK];

CI_lowerR=theta_boot(1,2)-1.96*StandardError_boot(1,2);
CI_upperR=theta_boot(1,2)+1.96*StandardError_boot(1,2);
CI_R=[CI_lowerR CI_upperR];

CI_lowerX=theta_boot(1,3)-1.96*StandardError_boot(1,3);
CI_upperX=theta_boot(1,3)+1.96*StandardError_boot(1,3);
CI_X=[CI_lowerX CI_upperX];

CI_ALL=[CI_K ; CI_R ; CI_X] % CI=[theta-1.96se theta+1.96se]

toc                         % computation time end
```

[Bootstrap_histogram]

```
%%% Bootstrap parameter distributions corresponding to 5% noise with CV
%%%

figure(1)  % for 'K'
hist(theta1(:,1))
h = findobj(gca,'Type','patch');
h.FaceColor = [1 0.7 0.7];
h.EdgeColor = 'w';
title('Distribution for K, nl=0.05')

figure(2)  % for 'r'
hist(theta1(:,2))
h = findobj(gca,'Type','patch');
h.FaceColor = [1 0.7 0.7];
h.EdgeColor = 'w';
title('Distribution for r, nl=0.05')

figure(3)  % for 'x0'
hist(theta1(:,3))
h = findobj(gca,'Type','patch');
h.FaceColor = [1 0.7 0.7];
h.EdgeColor = 'w';
title('Distribution for x_0, nl=0.05')
```

[LogisticModel.m]

```
function y = LogisticModel(theta,t)

y = theta(1)./(1+((theta(1)./theta(3))-1)*exp(-theta(2)*t));
```

[Logisticgraph.m]

```
theta=[17.5 0.7 0.1];
t=linspace(0,25,50);
n=length(t);
y=LogisticModel(theta,t);
plot(t,y,'r')
title('Logistic Curve with K=17.5, r=0.7 and x_0=0.1')
xlabel('Time')
ylabel('Model')
grid on
```

[Noisegraph.m]

```
clear all
clc

t=linspace(0,25,50);                    % time variable
theta=[17 0.6 0.1];                  % initial value of theta
theta_0=[17.5 0.7 0.1];               % true value of theta = (K, r, x0)
logis=LogisticModel(theta,t);
p=3;                                % the dimension of parameter space
n=length(t);                         % the length of the vector t
m=1000;                             % the number of bootstrap sample

lowb=[0 0 0];                        % lower bound of parameter space
uppb=[inf inf inf];                   % upper bound of parameter space

iterset=optimset('TolX',1.0e-4,'MaxFunEvals',10000);

y(1,:) = LogisticModel(theta_0,t);       % y is a value of random process
Y.
Y(1,:) = y(1,:)+(1/20)*randn(1,n);

plot(t,y,'r')
title('Logistic Curve with K=17.5, r=0.7 and x_0=0.1')
xlabel('Time')
ylabel('Model')
grid on
hold on
plot(t,Y(1,:),'*')
legend('Logistic curve','Noise points','Location','southeast')
hold off
```

[Table5_95CI.m]

```
%%%Table 5. 95%CI difference%%%

clear all
clc

s=linspace(0,25,1001);
subplot(1,2,1);
scatter(s',theta1(:,1),'b.'); xlim([0 25]); ylim([16.9 18.1]);
hold on
line([0 25],[17.053121 17.053121],'Color','red','LineStyle','--')
line([0 25],[18.000990 18.000990],'Color','red','LineStyle','--')
xlabel('Time');
ylabel('K_{boot}');
title('n=50 in Table 5');
subplot(1,2,2);
scatter(s',theta1(:,1),'b.'); xlim([0 25]); ylim([16.9 18.1])
line([0 25],[17.401983 17.401983],'Color','red','LineStyle','--')
line([0 25],[17.627816 17.627816],'Color','red','LineStyle','--')
xlabel('Time');
ylabel('K_{boot}');
title('n=1000 in Table 5');
legend('K_{boot}','95%CI');
hold off
```

[Asymptotic_CVdata.m]

```
%%%%% Asymptotic Theory using OLS for CV %%%%%

clear all
clc
format long

tic % computation times - start

% Setting the variables
t=linspace(0,25,50);
theta=[17 0.6 0.09];
theta_0=[17.5 0.7 0.1];
p=3; n=length(t);
lowb=[0 0 0]; uppb=[inf inf inf];
iterset=optimset('TolX',1.0e-4,'MaxFunEvals',10000);

y                    =                    LogisticModel(theta_0,t);

Y = y+(0.1)*randn(1,n);

residual=sqrt(n/(n-p)).*(Y-y);
r=randsample(residual,n,'true');

% OLS estimates of Asymptotic theory
theta_hat=lsqcurvefit(@LogisticModel,theta,t,Y,lowb,uppb,iterset);

%%% Setting sensitivity Matrix and Covariance %%%

% initial conditions
y0=[theta_hat(3);0;0];

% Sensitivity Matrix computations
[t, senMatrix]=ode45(@sensitivityEquation,t,y0,[],theta_hat);
```

```
%Covariance Matrix
sigma2=sum((y-LogisticModel(theta_hat,t)').^2)*(1/(n-p));
CovMatrix=sigma2*inv(senMatrix'*senMatrix);

% The results of algorithm
theta_hat
StandardError=sqrt(diag(CovMatrix))'


% 95% confidence intervals from Asymptotic estimates
CI_lowerK=theta_hat(1)-1.96*StandardError(1);
CI_upperK=theta_hat(1)+1.96*StandardError(1);
CI_K=[CI_lowerK CI_upperK];

CI_lowerR=theta_hat(2)-1.96*StandardError(2);
CI_upperR=theta_hat(2)+1.96*StandardError(2);
CI_R=[CI_lowerR CI_upperR];

CI_lowerX=theta_hat(3)-1.96*StandardError(3);
CI_upperX=theta_hat(3)+1.96*StandardError(3);
CI_X=[CI_lowerX CI_upperX];

CI_ALL=[CI_K ; CI_R ; CI_X] % CI=[theta-1.96se theta+1.96se]

toc % computation times - end
```

[Asymptotic_NCVdata.m]

```
%%%%% Asymptotic Theory using GLS for NCV %%%%%

clear all
clc
format long

tic % computation times - start

% Setting the variables
t=linspace(0,25,50);
theta=[17 0.6 0.09];
theta_0=[17.5 0.7 0.1];
p=3; n=length(t);

% Models
y=LogisticModel(theta_0,t);
Y=y+(1+(0.1))*randn(1,n);

Weights=1./(y.^2); % initial value of weights
residuesum = @(x) sum(Weights.*(Y-LogisticModel(x,t)).^2);

% GLS estimates of Asymptotic theory
theta_hat=fminsearch(residuesum,theta);

%%% Setting sensitivity Matrix and Covariance %%%

% initial conditions
y0=[theta_hat(3);0;0];
[t,y_hat]=ode45(@LogisticODE,t,theta_hat(3),[],theta_hat);
Weights_hat=1./(y_hat'.^2);

% Sensitivity Matrix computations
[t, senMatrix]=ode45(@sensitivityEquation,t,y0,[],theta_hat);
```

```
%Covariance Matrix
sigma2=sum(Weights_hat.*(y-LogisticModel(theta_hat,t)').^2).*(1/(n-p))
;
WeightMatrix=inv(diag(Weights_hat)); % WeightsMatrix (W^-1)^-1
CovMatrix=sigma2*inv(senMatrix'*WeightMatrix*senMatrix);

% The results of algorithm
theta_hat
StandardError=sqrt(diag(CovMatrix))'


% 95% confidence intervals from Asymptotic estimates
CI_lowerK=theta_hat(1)-1.96*StandardError(1);
CI_upperK=theta_hat(1)+1.96*StandardError(1);
CI_K=[CI_lowerK CI_upperK];

CI_lowerR=theta_hat(2)-1.96*StandardError(2);
CI_upperR=theta_hat(2)+1.96*StandardError(2);
CI_R=[CI_lowerR CI_upperR];

CI_lowerX=theta_hat(3)-1.96*StandardError(3);
CI_upperX=theta_hat(3)+1.96*StandardError(3);
CI_X=[CI_lowerX CI_upperX];

CI_ALL=[CI_K ; CI_R ; CI_X] % CI=[theta-1.96se theta+1.96se]

toc % computation times - end
```

[LogisticODE.m]

```
function dxdt = LogisticODE(t,x0,theta)
% initial value of x(0) = x0 = theta(3)
dxdt=theta(2).*x0.*(1-(x0./theta(1)));
```

[sensitivityEquation.m]

```
function seq=sensitivityEquation(t,x0,theta)
seq=[theta(2).*(1-(2*x0(1)./theta(1))) - theta(2)*(x0(1)*theta(1)).^2;
    theta(2).*(1-(2*x0(1)./theta(1)))*x0(2)                                +
x0(1).*(1-(x0(1)./theta(1)));
    theta(2).*(1-(2*x0(1)./theta(1)))*x0(2)];
```

참 고 문 헌

[1] Probability and statistics-Pearson Education(2012) / Morris H DeGroot_ Mark J Schervish.

[2] 비선형 회귀모형에서의 붓스트랩에 관한 연구 / 강철, 박종태 / 통계청『통계분석연구』제2권 제1호(’97. 봄) 143-160

[3] 회귀분석을 위한 부트스트래핑(bootstrapping)기법의 활용 / 심준섭(중앙대학교 공공정책학부)

[4] 회귀분석 강의노트(http://wolfpack.hnu.ac.kr/lecture/Regression/) / © 2010 SEHYUG KWON. ALL RIGHTS RESERVED

[5] Introduction to Mathematical Statistics 6th edition / Robert V. Hogg 외, Prentice Hall

[6] 회귀분석 제 3판 / 박성현 / 민영사