LM 최적화 알고리즘을 이용한 유리함수 모델의 데이터 피팅

A Data Fitting Technique for Rational Function Models Using the LM Optimization Algorithm

박 재 한*, 배 지 훈, 백 문 홍
(Jae-Han Park¹, Ji-Hun Bae¹, and Moon-Hong Baeg¹)

¹Korea Institute of Industrial Technology

Abstract: This paper considers a data fitting problem for rational function models using the LM (Levenberg-Marquardt) optimization method. Rational function models have various merits on representing a wide range of shapes and modeling complicated structures by polynomials of low degrees in both the numerator and denominator. However, rational functions are nonlinear in the parameter vector, thereby requiring nonlinear optimization methods to solve the fitting problem. In this paper, we propose a data fitting method for rational function models based on the LM algorithm which is renowned as an effective nonlinear optimization technique. Simulations show that the fitting results are robust against the measurement noises and uncertainties. The effectiveness of the proposed method is further demonstrated by the real application to a 3D depth camera calibration problem.

Keywords: LM (Levenberg-Marquardt) algorithm, nonlinear optimization, rational function model

I. 서론

데이터 피팅 문제는 어떠한 일련의 데이터 집합에 대하여 이 데이터들의 분포를 잘 표현할 수 있는 수학적 함수를 찾는 것을 말한다. 통계학 분야에서 데이터 피팅은 회귀분석(regression analysis)의 형태로서 수학적으로 모델을 정의할 수 없는 경우에 무작위적 오차를 포함하는 측정 데이터를 이용하여 이를 표현할 수 있는 적합한 함수를 찾도록 수행되며, 이는 여러 분야에서 유용하게 활용된다. 회귀모델은 회귀분석에서 피팅을 수행하는 대상 함수를 말하는데 이 논문에서는 여러 회귀 모델 중에서 유리함수를 이용한 데이터 피팅 문제를 다루고자 한다. 데이터 피팅 과정은 입력 데이터들의 분포와 회귀모델 함수와의 유사성을 정의한 성능지표 함수 또는 비용함수에 대한 최적화를 수행하는 것이다. 유리함수 모델의 피팅 문제에서는 비용함수가 최적화 파라미터에 대하여 선형 관계가 아니므로 비선형 최적화 기법을 적용해야 한다.

최적화 기법에 대해서는 많은 연구가 있었는데 크게 비용함수의 1차 미분을 이용한 최급강하(steepest descent)법, 공액구배(conjugate gradient)법과 2차 미분을 이용하는 뉴턴(newton)법, 의사 뉴턴(quasi-newton)법 등이 있으며 특히 비선형 최소 제곱 최적화 문제에는 뉴턴법에 기반을 둔 가우스-뉴턴법과 구배(gradient) 벡터, 헤시안(hessian) 행렬을 같이 이용하는 LM (Levenberg-Marquardt)법이 많이 적용되고

있다[1]. 이 논문에서는 자코비안 행렬의 특이점 문제를 극복하고 최적화 수렴속도가 뛰어나 비선형 최적화 문제에 많이 적용되는 LM 알고리즘을 활용하고자 하며 이를 이용한 유리함수 모델의 데이터 피팅 방법을 제시하고자 한다.

II. 데이터 피팅 문제 및 회귀 모델

1. 최소 놈(norm) 피팅 문제

데이터 피팅 문제는 식 (1)과 같이 주어진 k 개의 데이터 군(群)에 대하여 이의 분포와 가장 가까운 함수 $f(f \in F)$ 를 찾는 것을 말한다[2]. 여기서 F 는 데이터 피팅에 대한 후보 함수의 집합을, D 와 R 은 함수 f 의 정의역과 치역을 각각 의미한다.

$$(x_1, y_1), (x_2, y_2), \dots, (x_k, y_k) \quad (x_i \in D, y_i \in R) \quad (1)$$

데이터 군의 분포와 함수가 가깝다는 것은 식 (2)와 같이 입력 x 의 함수 값과 출력 y 간의 오차 e 의 놈(norm)이 작다는 의미이며 따라서 주어진 데이터와 가장 가까운 함수를 찾는 문제는 식 (2)의 l_p -놈이 최소화되는 함수 f 를 찾는 것이다.

$$\|e\|_p = \left[\sum_{i=1}^k |y_i - f(x_i)|^p \right]^{\frac{1}{p}} \quad (2)$$

일반적으로 많이 사용되는 최소 제곱(least-square) 피팅 문제는 특히 l_2 -놈이 최소화되는 함수를 찾는 문제로서 식 (3)과 같이 표현된다.

$$\text{minimize } \sum_{i=1}^k (y_i - f(x_i))^2 \quad (3)$$

* 책임저자(Corresponding Author)

논문접수: 2011. 5. 20., 수정: 2011. 6. 5., 채택확정: 2011. 6. 20.

박재한, 배지훈, 백문홍: 한국생산기술연구원

(hans1024@kitech.re.kr/joseph@kitech.re.kr/mhbaeg@kitech.re.kr)

그림 1. l_2 -놈(실선)과 l_∞ -놈(점선) 피팅 결과[2].

Fig. 1. Fitting results using l_2 -norm(solid) and l_∞ -norm(dashed).

그림 1은 동일한 데이터 분포에 대하여 l_2 -놈과 l_∞ -놈을 최소화하는 함수를 찾은 결과를 보여준다. 그림에서 작은 원은 측정 데이터를 실선은 오차에 대한 면적을 의미하는 l_2 -놈을 최소화하는 함수 그리고 점선은 오차의 최대 크기인 l_∞ -놈을 최소화하는 함수를 각각 나타내며, 최소화하고자 하는 척도에 따라 동일한 데이터에 대해서도 서로 다른 결과를 도출함을 알 수 있다.

2. 회귀 모델(regression models)

데이터 피팅 문제와 밀접한 관련이 있는 통계학 쪽에서는 피팅 후보 함수는 회귀 모델에서 모든 파라미터에 대한 함수들이 그 대상이 되며 따라서 후보 함수의 집합 F 는 식 (4)와 같이 나타낼 수 있다. 여기서 $\theta = [\theta_1, \theta_2, \dots, \theta_m]^T$ 는 m 차원의 파라미터 벡터이다.

$$F = \{f(\theta, x) \mid \theta \in R^m\} \quad (4)$$

특히 통계학 관련 분야에서 여러 회귀 모델에 대한 연구가 이루어지고 있지만, 일반적으로 다음과 같이 크게 다항식, 삼각함수, 유리함수 및 지수함수의 4가지 모델로 구분할 수 있다[3].

- Polynomial models: $f(\theta, x) = \sum_{i=1}^m \theta_i x^{i-1}$

- Trigonometrical models:

$$f(\theta, x) = \theta_0 + \sum_{i=1}^m (\theta_i \cos(ix)) + \sum_{i=1}^m (\theta_{m+i} \sin(ix))$$

- Rational models:

$$f(\theta, x) = \frac{P(\theta_1, x)}{Q(\theta_2, x)} \quad \left(\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} \right)$$

$$P(\theta_1, x) = \sum_{j=1}^d \theta_j x^{j-1}, \quad Q(\theta_2, x) = \sum_{j=1}^{m-d} \theta_{d+j} x^{j-1}$$

- Exponential models: $f(\theta, x) = \sum_{i=1}^m \theta_i e^{-\theta_{m+i} x}$

위의 회귀 모델에서 다항식, 삼각함수 모델은 함수를 파라미터 벡터와 회귀 모델의 데이터 벡터로 구분하여 두 벡

터의 곱으로 표현할 수 있는 선형 모델로서 특히 최소 제곱 피팅 문제에서는 그 해가 닫힌 해(closed-form) 형태로 바로 구할 수 있다. 유리함수와 지수함수는 두 벡터의 곱으로 표현할 수 없는 비선형 모델로서 최적의 함수를 찾기 위해서는 비선형 최적화 기법이 요구된다.

이 논문에서는 유리함수 모델에 대한 피팅 문제를 다루고자 한다. 유리함수 모델은 분모 다항식에 대한 제약조건과 적합한 차수를 선택하기가 애매한 단점이 있으나, 다양한 형태의 곡을 표현이 가능하고 뛰어난 내삽(interpolation) 및 외삽(extrapolation) 성능을 나타내며 다른 비선형 함수에 비해 비교적 다루기가 용이한 장점이 있어[4,5] 다양한 응용에서 유용하게 적용될 수 있을 것이다.

III. LM 최적화 기반 유리함수 피팅

1. LM (Levenberg-Marquardt) 최적화 알고리즘

LM 알고리즘[6]은 비선형 최적화 문제에 많이 사용되는 대표적인 방법이다. LM 알고리즘은 식 (5)와 같이 구배법과 뉴턴 최적화 방법을 μ 를 통해 결합함으로써 빠르고 정확한 최적해를 얻는 것을 목표로 한다. 여기서 $g(\theta)$ 는 목적함수 $L(\theta)$ 의 구배 벡터이며 $F(\theta)$ 는 헤시안 행렬을 α 는 최적화 변수 갱신 이득을 의미한다.

$$\begin{aligned} \theta_{k+1} &= \theta_k - \alpha_k (F(\theta_k) + \mu_k I)^{-1} g(\theta_k) \\ g(\theta_k) &= \nabla L(\theta_k), \quad F(\theta_k) = \nabla^2 L(\theta_k) \end{aligned} \quad (5)$$

그림 2(a)는 구배법에 의한 최적화 과정을 나타낸 것으로 최적화 변수의 갱신 크기가 제한되므로 초기치가 최적값에 멀리 떨어진 경우에는 최적해를 구할 때 많은 반복과

(a) Gradient method.

(b) Newton's method.

그림 2. 구배법 및 뉴턴법의 최적화 과정[8].

Fig. 2. Optimization processes of Gradient /Newton's methods.

정이 요구된다. 또한 그림 2(b)와 같이 목적함수를 2차(quadratic) 형태로 근사화하여 이의 최적해를 구하며 최적화 변수를 찾아가는 뉴턴법은 초기치가 최적값과 먼 위치에서 시작해도 빠르게 최적값 근처로 이동하는 장점이 있으나 최적값 근방에서는 헤시안 행렬의 변화가 매우 적어서 수렴속도가 느려지는 것으로 알려져 있다[1].

LM 알고리즘에서는 μ 를 통해 두 가지 방법이 결합되어 있는데, μ 가 작은 값이면 뉴턴법과 유사하고 반대로 매우 큰 값이면 구배법과 유사해진다. 일반적으로 최적화 초기에는 μ 를 작은 값에서 시작해서 뉴턴법의 특성이 크게 나타나도록 하여 최적값 근처로 빠르게 이동하도록 하고, 점점 μ 를 증가하여 구배법의 특성이 크게 나타나도록 함으로써 최적값 근처에서 일정한 수렴속도를 보장할 수 있도록 조절한다[7].

2. LM 알고리즘을 이용한 유리함수 모델 피팅

유리함수 모델에서 주어진 데이터 군과 가장 가까운 함수를 찾는 피팅 문제는 식 (6)과 같이 표현할 수 있다. 여기서 $f(\theta, x)$ 는 분자 다항식 $P(\theta, x)$ 와 분모 다항식 $Q(\theta, x)$ 의 비로 표현된 유리함수이며 θ 는 $m+n$ 차원의 파라미터 벡터이다. 또한 $r_i(\theta, x_i)$ 는 각 데이터에 대한 잔차(residual)를 나타내며 목적함수 $L(\theta)$ 를 모든 데이터에 대한 잔차의 제곱합으로 선정함으로써 최소 제곱 피팅 문제가 된다.

$$(x_1, y_1), (x_2, y_2), \dots, (x_k, y_k) \quad (x_i \in D, y_i \in R)$$

$$\text{minimize } L(\theta) = \sum_{i=1}^k (r_i(\theta, x_i))^2 \quad [r_i(\theta, x_i) = y_i - f(\theta, x_i)] \quad (6)$$

$$f(\theta, x) = \frac{P(\theta, x)}{Q(\theta, x)} = \frac{\sum_{j=1}^m \theta_j x^{j-1}}{\sum_{j=1}^n \theta_{m+j} x^{j-1}} \quad \theta \in R^m \times R^n$$

위의 문제에 LM 알고리즘을 적용하기 위해서는 목적함수 $L(\theta)$ 의 구배 벡터와 헤시안 행렬을 구해야 하는데 이를 위하여 잔차 벡터(residual vector) $r(\theta)$ 를 식 (7)과 같이 정의하면 목적함수는 식 (8)의 형태가 된다.

$$r(\theta) = [r_1(\theta, x_1) \cdots r_k(\theta, x_k)]^T \quad (7)$$

$$L(\theta) = \sum_{i=1}^k (r_i(\theta, x_i))^2 = r(\theta)^T r(\theta) \quad (8)$$

여기서 구배 벡터는 식 (9)와 같이 구할 수 있는데 이는 식 (10)과 같이 자코비안 행렬 $J(\theta)$ 와 잔차 벡터 $r(\theta)$ 의 곱으로 나타낼 수 있다.

$$(\nabla L(\theta))_j = \frac{\partial L(\theta)}{\partial \theta_j} = 2 \sum_{i=1}^k r_i(\theta, x_i) \frac{\partial r_i(\theta, x_i)}{\partial \theta_j}$$

$$\nabla L(\theta) = \begin{bmatrix} (\nabla L(\theta))_1 \\ \vdots \\ (\nabla L(\theta))_{m+n} \end{bmatrix} = \begin{bmatrix} 2 \sum_{i=1}^k r_i(\theta, x_i) \frac{\partial r_i(\theta, x_i)}{\partial \theta_1} \\ \vdots \\ 2 \sum_{i=1}^k r_i(\theta, x_i) \frac{\partial r_i(\theta, x_i)}{\partial \theta_{m+n}} \end{bmatrix} \quad (9)$$

$$= 2 \begin{bmatrix} \frac{\partial r_1(\theta, x_1)}{\partial \theta_1} & \cdots & \frac{\partial r_k(\theta, x_k)}{\partial \theta_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_1(\theta, x_1)}{\partial \theta_{m+n}} & \cdots & \frac{\partial r_k(\theta, x_k)}{\partial \theta_{m+n}} \end{bmatrix} \begin{bmatrix} r_1(\theta, x_1) \\ \vdots \\ r_k(\theta, x_k) \end{bmatrix}$$

$$g(\theta) = \nabla L(\theta) = 2 J(\theta)^T r(\theta)$$

$$J(\theta) = \begin{bmatrix} \frac{\partial r_1(\theta, x_1)}{\partial \theta_1} & \cdots & \frac{\partial r_1(\theta, x_1)}{\partial \theta_{m+n}} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_k(\theta, x_k)}{\partial \theta_1} & \cdots & \frac{\partial r_k(\theta, x_k)}{\partial \theta_{m+n}} \end{bmatrix} \quad (10)$$

헤시안 행렬은 식 (11)과 같이 구할 수 있는데 이는 $J(\theta)^T J(\theta)$ 와 $S(\theta)$ 로 나눌 수 있으며 $S_{k,j}$ 로 구성된 $S(\theta)$ 행렬은 모든 요소가 $J(\theta)^T J(\theta)$ 행렬에 비해 매우 작기 때문에 무시할 수 있게 된다. 따라서 식 (12)와 같이 헤시안 행렬도 자코비안 행렬의 곱으로 근사적으로 표현할 수 있다.

$$\frac{\partial^2 L(\theta)}{\partial \theta_k \partial \theta_j} = \frac{\partial}{\partial \theta_k} \left(\frac{\partial L(\theta)}{\partial \theta_j} \right)$$

$$= \frac{\partial}{\partial \theta_k} \left(2 \sum_{i=1}^k r_i(\theta, x_i) \frac{\partial r_i(\theta, x_i)}{\partial \theta_j} \right)$$

$$= 2 \sum_{i=1}^k \left(\frac{\partial r_i(\theta, x_i)}{\partial \theta_k} \frac{\partial r_i(\theta, x_i)}{\partial \theta_j} + r_i(\theta, x_i) \frac{\partial^2 r_i(\theta, x_i)}{\partial \theta_k \partial \theta_j} \right)$$

$$= 2 (J(\theta)^T J(\theta) + S(\theta)) \quad (11)$$

$$\left(S_{k,j} = r_i(\theta, x_i) \frac{\partial^2 r_i(\theta, x_i)}{\partial \theta_k \partial \theta_j} \right)$$

$$F(\theta) = \nabla^2 L(\theta) = 2 (J(\theta)^T J(\theta) + S(\theta)) \approx 2 J(\theta)^T J(\theta) \quad (12)$$

식 (10)과 식 (12)의 결과로부터 최소 제곱 피팅 문제에 대한 LM 알고리즘은 식 (13)과 같이 정리되며 이의 실현을 위해서는 오직 자코비안 행렬 $J(\theta)$ 만이 요구된다.

$$\theta_{k+1} = \theta_k - \alpha_k (F(\theta_k) + \mu_k I)^{-1} g(\theta_k)$$

$$\Downarrow$$

$$\theta_{k+1} = \theta_k - \alpha_k (J(\theta_k)^T J(\theta_k) + \mu_k I)^{-1} J(\theta_k)^T r(\theta_k) \quad (13)$$

여기서 모델 $f(\theta, x)$ 가 유리함수이므로 잔차 $r_i(\theta, x_i)$ 에 대한 편미분은 식 (14)와 같으며 따라서 일반화된 유리함수 모델의 최소 제곱 피팅에 대한 목적함수의 자코비안 행렬은 식 (15)의 형태로 구할 수 있다.

$$\frac{\partial r_i(\theta, x_i)}{\partial \theta_j} = \frac{\partial}{\partial \theta_j} (y_i - f(\theta, x_i))$$

$$= - \frac{\partial}{\partial \theta_j} f(\theta, x_i)$$

$$= \begin{cases} - \frac{(x_i)^{j-1}}{Q(\theta, x_i)} & (j \leq m) \\ \frac{P(\theta, x_i)(x_i)^{j-m-1}}{Q(\theta, x_i)^2} & (j > m) \end{cases} \quad (14)$$

그림 3. Levenberg-Marquardt 알고리즘의 최적화 과정.

Fig. 3. Optimization processes of the LM algorithm.

$$J = \begin{bmatrix} \frac{1}{Q(\theta, x_1)} \cdots \frac{(x_1)^{m-1}}{Q(\theta, x_1)} & \frac{P(\theta, x_1)}{Q(\theta, x_1)^2} \cdots \frac{P(\theta, x_1)(x_1)^{n-1}}{Q(\theta, x_1)^2} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{1}{Q(\theta, x_k)} \cdots \frac{(x_k)^{m-1}}{Q(\theta, x_k)} & \frac{P(\theta, x_k)}{Q(\theta, x_k)^2} \cdots \frac{P(\theta, x_k)(x_k)^{n-1}}{Q(\theta, x_k)^2} \end{bmatrix} \quad (15)$$

그림 3은 LM 알고리즘을 통하여 데이터 피팅이 진행되는 과정을 나타낸 것이다. (*)로 표기된 정보가 입력 데이터로서 원시 데이터에 가우시안 노이즈를 인가한 것이며 이를 이용하여 LM 최적화를 수행하였다. 파라미터 초기값은 랜덤하게 선정되는데 이로 인해 초기 함수가 주어진 데이터와 비교하여 많은 차이가 나더라도 반복 과정을 거치면서 함수의 그래프가 데이터의 경향에 맞춰지며 피팅이 수행되는 것을 확인할 수 있다.

3. 데이터의 가중치를 고려한 피팅 방법

주어진 데이터의 일부에 측정 신뢰성 문제가 있다거나 또는 어떤 데이터에 대해서는 피팅을 잘 맞추는 것이 중요한데 비해 다른 데이터는 상대적으로 덜 중요한 등의 경우에는 데이터의 중요도에 따른 가중치를 부가하여 최적화를 수행할 필요가 있다[9]. 이를 위하여 주어진 데이터에 각각 가중치 w_i 를 부여하여 이를 고려한 목적함수를 식 (16)과 같이 나타낼 수 있다. 여기서 가중치 w_i 는 0보다 큰 양의 실수가 되어야 한다.

$$\begin{aligned} & (x_1, y_1, w_1), (x_2, y_2, w_2), \dots, (x_k, y_k, w_k) \\ & (x_i \in D, y_i \in R, w_i \in R_{++}) \end{aligned} \quad (16)$$

$$\text{minimize } L(\theta) = \sum_{i=1}^k (w_i r_i(\theta, x_i))^2$$

식 (16)에 따라 가중 잔차 벡터(weighted residual vector) $r_W(\theta)$ 및 목적함수는 식 (17)과 같이 되며 구배 벡터와 헤시안 행렬도 식 (18)과 같이 정리할 수 있다.

$$r_W(\theta) = \begin{bmatrix} w_1 r_1(\theta, x_1) \\ \vdots \\ w_k r_k(\theta, x_k) \end{bmatrix} = W r(\theta), \quad \left(W = \begin{bmatrix} w_1 & & 0 \\ & \ddots & \\ 0 & & w_k \end{bmatrix} \right) \quad (17)$$

(a) Residuals for non-weighted optimization.

(b) Residuals for weighted optimization.

그림 4. 가중치가 고려에 따른 최적화 결과 비교.

Fig. 4. Optimization results for weighted and non-weighted case.

$$\begin{aligned} L(\theta) &= r_W(\theta)^T r_W(\theta) = r(\theta)^T W^T W r(\theta) \\ &= r(\theta)^T Q r(\theta) \quad (Q = W^T W) \\ \begin{cases} g(\theta) = \nabla L(\theta) = 2 J(\theta)^T Q r(\theta) \\ F(\theta) = \nabla^2 L(\theta) \approx 2 J(\theta)^T Q J(\theta) \end{cases} \end{aligned} \quad (18)$$

따라서 식 (13)의 LM 알고리즘 식에서 가중치가 고려된 구배 벡터 및 헤시안 행렬을 적용하면 식 (19)의 가중치가 고려된 LM 알고리즘 식을 얻을 수 있다.

$$\theta_{k+1} = \theta_k - \alpha_k (J(\theta_k)^T Q J(\theta_k) + \mu_k I)^{-1} J(\theta_k)^T Q r(\theta_k) \quad (19)$$

그림 4는 가중치 고려에 따른 최적화 수행 결과를 피팅 오차로써 비교하여 나타내고 있다. (a)는 가중치를 고려하지 않은 피팅 결과이고 (b)는 가중치를 고려한 결과이다.

그림 4의 데이터는 x 가 증가할수록 y 에 오차가 크게 인가되는 경우에 대한 데이터로서, 오차를 작게 포함하는 데이터의 가중치는 높이고 상대적으로 큰 오차를 포함하는 데이터의 가중치는 낮추어 최적화를 수행한 결과이다. (a)의 가중치를 고려하지 않은 결과를 보면 큰 오차를 포함하는 뒤쪽의 데이터가 고려되어 비교적 정확한 앞쪽의 데이

터에서도 피팅 오차가 존재하는 것을 알 수 있다. 그러나 (b)의 결과에서는 뒤쪽에서 피팅 오차가 조금 더 클 수 있으나 비교적 정확한 앞쪽에서는 피팅 오차가 많이 감소했음을 알 수 있다. 따라서 주어진 데이터의 중요도나 불확실성에 대한 정보를 파악할 수 있다면 이에 대한 가중치를 고려한 LM 알고리즘을 통하여 실제로 더 가까운 피팅 함수를 찾을 수 있을 것으로 예상된다.

4. 유리함수 모델의 최적화 제약조건 및 최적 차수 선택

앞서 언급한 바와 같이 데이터 피팅 문제에 회귀모델로서 유리함수를 적용하는 것은 많은 장점을 가지고 있다. 그러나 유리함수의 최적화 문제에서는 분모다항식에 대한 제약조건을 반드시 고려해야 한다. 분모다항식 $Q(\theta, x)$ 가 x 의 정의역 D 내에서 근을 가지게 되면 함수 $f(\theta, x)$ 를 정의할 수 없으므로 최적화 과정에서 x 의 정의역 내에서 분모다항식의 근이 존재하지 않도록 제약조건을 두어야 한다.

그림 5는 유리함수 피팅 최적화에 대한 제약조건을 나타낸 것이다. 복소근을 포함한 분모다항식 $Q(\theta, x)$ 의 근이 존재해서는 안되는 영역을 x 의 정의역에 실수축 여유폭 (margin) M_{Re} , 허수축 여유폭 M_{Im} 을 각각 인가하여 그림에서의 회색 영역과 같이 정의하였다. 최적화를 진행하면서 분모다항식의 근을 조사하여 그림의 회색 금지 영역에 근이 존재하는 경우에는 최적화를 중단하고 그렇지 않으면 최적화 파라미터를 갱신하며 알고리즘을 계속 수행한다. 식 (20)은 분모다항식에 대한 제약조건을 고려한 최적화 알고리즘을 나타내고 있다.

$$\begin{aligned} \theta_{tmp} &= \theta_k - \alpha_k (J(\theta_k)^T J(\theta_k) + \mu_k I)^{-1} J(\theta_k)^T r(\theta_k) \\ C &= \text{roots}(Q(\theta_{tmp}, x)) \\ \text{if } (x_{\min} - M_{Re} \leq \text{Re}(C) \leq x_{\max} + M_{Re}) \\ &\quad \text{and } (|\text{Im}(C)| \leq M_{Im}) \\ &\quad \text{terminates optimization} \\ \text{else} \\ &\quad \theta_{k+1} = \theta_{tmp} \\ \text{end} \end{aligned} \quad (20)$$

또한 유리함수의 피팅 최적화에 있어서는 분모, 분자의 적합한 차수를 결정하는 문제도 존재한다. 이를 위하여 이 논문에서는 분모, 분자의 차수를 동일하게 함으로써 차수 결정을 위한 탐색공간을 2차원에서 1차원으로 단순화하고 저차부터 시작해 차수를 늘려가면서 그에 따른 성능 지표

그림 5. 유리함수 피팅 최적화에 대한 제약조건.

Fig. 5. Constraints for rational function fitting optimization.

를 비교함으로써 최적 차수를 결정하는 방법을 제안한다. 분모, 분자의 차수를 동일하게 하면 차수 결정에 대한 복잡성도 현저하게 줄어드는 장점도 있지만, 많은 경우에 있어서 앞서 언급한 유리함수의 분모다항식 근에 대한 제약조건을 만족하면서 최적화가 수행되는 것을 실험적으로 확인할 수 있었다. 최적 차수를 결정하는 조건은 식 (21)과 같이 정의할 수 있는데 차수의 증가에 대한 피팅 성능을 나타내는 제곱 오차의 변화가 감소하여 한계치 γ 보다 작으면 그 이전의 차수를 최적 차수로 결정할 수 있다.

$$\text{Optimal order criterion: } \frac{\Delta L(\theta)}{\Delta \text{order}} < \gamma \quad (21)$$

그림 6은 차수의 증가에 따른 최적화된 피팅 함수의 결과와 그에 대한 최적 차수 결정을 위한 성능지표 결과를 나타내고 있다. 차수의 탐색 공간은 최소 2차부터 시작해 11차까지 증가하면서 그에 따른 피팅 결과를 비교하였으며 (b)의 결과에 따르면 분모, 분자 6차의 유리함수가 최적의 피팅 함수임을 알 수 있다. (a)의 결과를 보면 2차에서 6차까지는 피팅 결과의 변화가 많이 나타나며 그 이후로는 차수의 증가가 피팅 결과의 변화에 영향을 미치지 않는 것이 확인된다. 또한 차수가 증가하여 미소한 성능의 변화가 있더라도 고차항에 의한 연산의 복잡성을 고려하면 낮은 차수의 유리함수가 최적의 피팅 함수로 선택되는 것이 타당할 것이다.

(a) Plot of fitting result functions.

(order = Green: 2~3, Cyan: 4~6, Blue: 7~9, Red: 10~)

(b) Performance index graph.

그림 6. 차수의 증가에 대한 유리함수의 피팅 결과.

Fig. 6. Fitting results of rational function for increasing of order.

IV. 모의실험 및 응용 예

1. 유리함수를 이용한 피팅 모의실험

다양한 곡률에 대한 유리함수 모델의 피팅 성능을 확인하기 위하여 식 (22)와 같이 분모, 분자 10차의 유리함수 모델을 랜덤하게 생성하여 분모, 분자 5차의 유리함수 모델로 피팅을 수행하였다. 입력 데이터는 25개의 샘플링된 모델함수의 값에 $[\mu=0, \sigma=\sigma_{test}]$ 의 가우시안 잡음을 인가하여 생성하였으며 피팅 성능의 비교를 위하여 가장 일반적인 방법인 다항식(polynomial) 모델과 그 결과를 비교하였다.

input data: $(x_i, y_i) \quad i=1 \dots 25$

$$y_i = f(\theta, x_i) + \xi \quad (\xi: \text{Gaussian noise} [\mu=0, \sigma=\sigma_{test}])$$

$$f(\theta, x) = \frac{\sum_{j=1}^{10} \theta_j x^{j-1}}{\sum_{j=1}^{10} \theta_{10+j} x^{j-1}} \quad (\theta \in R^{20}) \quad (22)$$

그림 7은 랜덤하게 생성된 함수의 데이터 중 선별한 두 패턴과 그 피팅 결과를 나타낸 것으로 이 때 가우시안 잡음의 σ_{test} 를 0.07과 0.1로 각각 인가하였다. 그림에서 (o)는 원시 데이터를 (+)는 잡음이 포함된 데이터를 나타내며 (*)는 유리함수로 피팅된 결과를 (x)는 10차 다항식으로 피팅된 결과를 각각 나타낸다. 유리함수 모델의 피팅 결과를 보

(a) $\sigma_{test} = 0.07$.

(b) $\sigma_{test} = 0.1$.

그림 7. 다양한 패턴과 측정 잡음에 대한 피팅 결과.

Fig. 7. Fitting results for various input patterns and noises.

면 비교적 큰 잡음을 포함한 데이터에 대해서도 원시 데이터의 경향에 잘 맞춰지며 피팅이 이루어진 것으로 확인되지만 다항식 모델은 잡음의 영향이 피팅 결과에 민감하게 반영되어 원시 데이터의 경향과 차이가 크게 나타난다. 다른 형태의 곡률에 대해서도 아래와 유사한 결과를 얻었는데 이러한 결과로부터 유리함수 모델을 이용하여 다양한 형태의 곡률 표현과 잡음에 강인한 피팅 수행이 가능함을 확인할 수 있었다.

입력 데이터에 포함된 잡음에 대한 영향을 파악하기 위하여 동일한 입력에 잡음의 강도 σ_{test} 를 높여가면서 유리함수와 다항식 모델에 대한 피팅을 수행하였는데 그림 8의 (a)~(f)에서 그 결과를 나타내고 있다. 결과로부터 다항식 모델은 σ_{test} 의 증가에 따라 나타나는 샘플링된 입력 데이터의 급격한 변화를 피팅에 반영하면서 그 이외의 구간에서는 오히려 오차가 더욱 크게 나타남을 알 수 있다.

그에 비해 유리함수 모델은 입력 데이터의 변화를 반영하기 보다는 데이터 전체의 경향에 맞추어 피팅이 이루어지는 것으로 보이며 따라서 데이터 이외의 구간에서도 오차가 크게 나타나지 않음을 알 수 있다. 이를 통해서 유리함수 모델을 이용한 최적화 결과는 앞서 언급했듯이 우수한 내삽(interpolation) 특성을 보임을 확인할 수 있다. 그림 9는 σ_{test} 의 증가에 따른 피팅 오차의 l_{∞} -norm을 나타낸 것으로 다항식 모델 오차인 (o)의 큰 변화에 비해 (*)의 유리함수 모델 오차는 변화량이 현저히 낮게 나타났다.

표 1은 식 (22)의 유리함수 모델에 대하여 무작위로 함수를 생성하고 σ_{test} 강도의 잡음을 인가하여 구성된 입력 데이터에 대하여 피팅 수행 중 실패한 횟수를 나타낸 것이다. 여기서 피팅 실패라는 것은 유리함수의 분모다항식 제약조건을 만족하지 못하여 최적화가 중단되거나 최적화 변수가 국부 최소점(local minimum)에 수렴하여 최적화를 수행한 후에도 피팅 오차가 크게 나타나는 경우를 의미한다. 모의실험은 각 σ_{test} 에 대하여 1,000번의 시행을 5회씩 수행하였으며 이에 대한 실험결과를 정리한 것이다. 결과로부터 σ_{test} 가 0.05 이하에서는 피팅이 적절하게 수행되는 것으로 보이나 σ_{test} 가 0.05를 넘어가면 피팅 실패가 발생할 가능성이 비교적 높은 것으로 나타난다. σ_{test} 가 0.05인 경우에 측정오차는 99.7%의 발생확률로 $\pm 0.15(3\sigma)$ 크기로 인가되며 그에 대한 데이터 분포는 그림 8(f)의 그래프를 통해 가늠할 수 있다. 위의 결과는 아무런 제약조건 없이 무작위로 생성한 함수에 대하여 랜덤 초기치에 의해 피팅 최적화를 수행한 것이므로 실제 응용에서는 피팅 오류가 발생하면 초기치를 바꾸거나 모델의 차수를 조정하는 등의 후처리를 통하여 최적의 피팅 함수를 찾을 수 있을 것이다.

2. 깊이감지 카메라의 거리 캘리브레이션 응용

이 논문에서 제시한 유리함수 모델의 피팅 방법을 깊이감지 카메라의 거리 캘리브레이션에 응용한 예를 소개한다. 깊이감지 카메라는 3차원 정보를 카메라처럼 실시간으로 획득할 수 있는 장치로 최근 로봇을 비롯한 다양한 분야에서 활용이 점점 확대되고 있다[10,11]. 그러나 깊이감지 카메라의 정보는 상대거리 정보로서 실제 공간 정보와 차이

그림 8. 잡음의 강도 변화에 대한 피팅 결과.

Fig. 8. Fitting results for change of noise strength.

그림 9. σ_{test} 증가에 따른 피팅 오차의 l_∞ -놈.Fig. 9. l_∞ -norm of fitting error for increasing of σ_{test} .

가 있으며 따라서 깊이감지 카메라로 3차원 공간 정보를 얻기 위해서는 거리에 대한 캘리브레이션이 필요하다.

캘리브레이션을 위하여 최소/최대 감지 거리인 0.5m에서 17.38m까지의 구간에 대하여 300개의 깊이감지 카메라 및 실제 거리데이터 집합을 실험을 통해 구하였다. 그림 10(a)의 (o)로 표시된 점들이 입력 데이터로 사용되는 300개의 실험 측정치를 나타낸다. 데이터의 분포를 살펴보면 어떠한 곡률을 가진 곡선 형태이므로 유리함수를 이용한 피팅에 적합할 것으로 보인다. 또한 가까운 거리에서는 측

표 1. 잡음 강도에 따른 피팅 실패 발생 횟수.

Table 1. # of fitting fail occurrences for various noise strength.

σ_{test} 실험 횟수	피팅 실패 발생 횟수 (1,000번 시행 중)						
	0.01	0.02	0.03	0.04	0.05	0.07	0.1
1	25	30	61	99	141	260	421
2	30	31	52	107	128	281	414
3	15	28	48	86	149	264	429
4	31	35	64	110	146	262	397
5	24	49	59	96	151	268	399
평균 발생률	2.5%	3.46%	5.68%	9.96%	14.3%	26.7%	41.2%

정 데이터의 변화가 작게 나타나지만 거리가 멀어질수록 데이터의 변화가 점점 커지게 되는데 이는 먼 거리에 대해서는 해상도와 정확도가 낮아지는 것을 의미한다. 따라서 정확도가 높은 가까운 거리에 대해서는 높은 가중치를 인가하고 정확도가 낮은 먼 거리에 대해서는 낮은 가중치를 인가하여 피팅을 수행하는 것이 바람직하다. 이를 반영한 가중치 벡터와 (o)로 표시된 입력데이터 군을 이용하여 유리함수 피팅을 수행하였으며 그 결과는 (a)의 실선으로 표시된 그래프이다. 그림 10(b)는 피팅 잔차 벡터를 나타낸 것으로 가까운 거리에서는 오차가 매우 작으나 먼 거리에서는 오차가 비교적 크게 피팅 되었음이 확인된다. 본 데이터에 대한 유리함수 모델의 최적 차수는 분모, 분자 5차(4차 다항식)로 결정되었으며 그에 대한 최적 파라미터

(a) Plots of measured data and fitting function.

(b) Plot of fitting residual.

그림 10. 깊이감지 카메라의 거리 피팅 결과.

Fig. 10. Distance fitting result for depth sensing camera.

는 식 (23)과 같다.

$$\text{Optimal parameter: } \theta^* = \begin{bmatrix} 452.704854 \\ -611.067608 \\ 255.253779 \\ -7.295045 \\ 7.345993 \\ -326.149609 \\ 588.445583 \\ -548.754315 \\ 340.178453 \\ -47.175180 \end{bmatrix} \quad (23)$$

그림 11은 최적 피팅 함수를 이용하여 거리 캘리브레이션을 수행한 후 깊이감지 카메라로 3차원 공간을 획득한 결과를 나타낸다. (a)는 3차원 스캔한 공간을 (b)는 깊이감지 카메라의 거리영상을 나타낸다. (c)는 거리영상 데이터로 3차원 정보를 구성하여 점군(point cloud)으로 나타낸 것인데 먼 거리의 데이터가 비슷한 깊이의 데이터로 모여서 3차원 공간정보가 제대로 구성되지 않고 있다. (d)는 캘리브레이션 된 거리정보를 이용하여 3차원 정보를 구성한 것인데 실제 공간을 잘 나타내고 있는 것이 확인된다.

(a) Scanning space.

(b) Depth image.

(c) 3D point cloud plot for raw depth data.

(d) 3D point cloud plot for calibrated depth data.

그림 11. 깊이감지 카메라를 이용한 3D 데이터 획득 결과.

Fig. 11. 3D acquisition result using depth sensing camera.

V. 결론

이 논문에서는 유리함수 형태의 회귀 모델을 이용한 데이터 피팅 방법으로서 비선형 최적화 기법인 LM (Levenberg-Marquardt) 알고리즘을 이용한 최적화 기법을 제안하였다. 이를 위하여 일반화된 유리함수의 자코비안 행렬을 도출하였으며 이를 LM 알고리즘에 적용하여 최적화를 수행함으로써 입력 데이터와 유리함수 모델의 피팅이 잘 이루어지는 것을 확인하였다. 또한 가중치가 고려된 LM 알고리즘식을 유도하여 피팅시 중요도를 부여하고 중요도가 높은 데이터의 피팅이 더 잘 이루어지도록 최적화 수행이 가능함을 보였다. 이와 함께 유리함수 모델의 최적화에 있어서 요구되는 제약조건과 알고리즘에서 이를 회피할 수 있는 방법을 제시하였으며 유리함수의 최적 차수 선택을 위한 성능지표와 조건을 제안하여 사용자의 간섭을 최소화한 자동화된 데이터 피팅 방법을 실현하고자 하였다.

실험결과로부터 유리함수 모델이 특히 인가되는 잡음 강도의 변화에 둔감하며 강인하게 최적함수를 추정하는 것으로 관찰되었다. 이는 곡률형태의 변화를 가지면서 측정 잡음이나 불확실성을 포함하는 데이터에서 그 경향을 추정하는데 유리함수 모델을 이용한 최적화 기법이 유용할 것으로 보인다. 또한 논문에서 제안한 방법을 깊이감지 카메라의 거리 캘리브레이션 문제에 적용하여 적합한 거리 캘리브레이션 결과를 얻었고 그에 따른 올바른 3차원 정보의 획득을 확인함으로써 실제 문제에서도 제안한 방법이 유용하게 활용될 수 있음을 보였다.

참고문헌

- [1] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer, New York, 2006.
- [2] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge Univ. Press, Cambridge, 2004.
- [3] V. B. Melas, *Functional Approach to Optimal Experimental Design*, Springer, New York, 2006.
- [4] T. P. Ryan, *Modern Regression Methods*, Wiley, Hoboken, 2009.
- [5] NIST, Engineering Statistics Handbook, <http://www.itl.nist.gov/div898/handbook/>.
- [6] D. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *SIAM Journal on Applied Mathematics*, vol. 11, no. 2, pp. 431-441, 1963.
- [7] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C++ 3rd Edition*, Cambridge Univ. Press, Cambridge, 2007.
- [8] K. P. Chong and S. H. Zak, *An Introduction to Optimizations*, Wiley, New York, 2001.
- [9] W. S. Cleveland and S. J. Devlin, "Locally weighted regression: an approach to regression analysis by local fitting," *Journal of the American Statistical Association*, vol. 83, pp. 596-610, 1988.
- [10] PrimeSense, <http://www.primesense.com/>.
- [11] Wikipedia-kinect, <http://en.wikipedia.org/wiki/Kinect>.

박재한

1998년 동아대학교 전자공학과 학사.
2000년 부산대학교 전자공학과 석사.
2010년 고려대학교 전기전자전파공학과 박사 수료. 2004년~현재 한국생산기술연구원 로봇융합연구그룹 선임연구원. 관심분야는 3차원 정보 처리, 인식기반 로봇 작업.

배지훈

1999년 명지대학교 전기전자공학과 학사. 2001년 동 대학원 석사. 2004년 Ritsumeikan 대학교 로봇공학학과 박사. 2009년~현재 한국생산기술연구원 로봇융합연구그룹 선임연구원. 관심분야는 양팔 로봇 작업, 로봇 손 제어.

백문홍

1982년 서울대학교 제어계측공학과 학사. 1984년 동 대학원 석사. 1995년 동경대학교 전기 전공 박사. 1996년~현재 한국생산기술연구원 로봇융합연구그룹 수석연구원. 관심분야는 로봇 비전, 3차원 공간 인지.