T.C. SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ BÖLÜMÜ

2025-2026 EĞİTİM-ÖĞRETİM YILI GÜZ DÖNEMİ MAT 217 - SAYISAL ANALİZ DERSİ

DÖNEM PROJESİ RAPORU

(Konu: Veri Merkezi Soğutma Sistemlerinde Enerji Verimliliği Optimizasyonu)

Hazırlayan (Öğrenci): Abdulrab Abdulrab B221200563

Dersi Veren Öğretim Üyesi: Dr. Öğr. Üyesi Fatma AKALIN

KASIM, 2025 SAKARYA

1. Problemin Tanımı

Proje Başlığı: Veri Merkezi (Data Center) Soğutma Sistemlerinde Enerji Verimliliği Optimizasyonu

Problemin Tanımı:

Modern veri merkezlerinde, sunucu raflarının (server racks) soğutulması kritik bir mühendislik problemidir. Fan hızının düşük olması durumunda sunucular aşırı ısınarak donanım arızalarına ve veri kaybına yol açar. Öte yandan, fan hızının gereğinden yüksek olması durumunda enerji tüketimi fan yasaları gereği kübik olarak artar ve işletme maliyetlerini gereksiz yere yükseltir.

Temel sorun; bu iki zıt durum (aşırı ısınma riski vs. yüksek enerji maliyeti) arasında dengeyi sağlayan ideal çalışma noktasının bulunmasıdır.

Mühendislik Hedefi:

Bir sunucu soğutma sisteminde, fan dönüş hızını (v) kontrol değişkeni olarak kullanarak, toplam maliyet fonksiyonunu (Enerji Maliyeti + Risk Maliyeti) minimize eden optimum fan hızını (v_opt) sayısal analiz yöntemleri ile belirlemektir.

Ölçülebilir Parametreler:

- Bağımsız Değişken: Fan Hızı, v (RPM veya % kapasite).

- Bağımlı Değişkenler:

Güç Tüketimi P(v)
Risk Maliyeti R(v).

- Kısıtlar: İşlemci sıcaklığı (T), belirlenen güvenlik sınırının (örneğin 80°C) altında kalmalıdır.

2. Fiziksel Modelin Oluşturulması

Problemin matematiksel modele dökülebilmesi için, gerçek hayat karmaşasından arındırılmış aşağıdaki fiziksel kabuller ve varsayımlar yapılmıştır:

1. Sistem Sınırları:

Analiz, tek bir sunucu kabini (Server Rack) için yapılacaktır. Ortam sıcaklığı (T_ortam) ve sunucu işlemci yükü sabit kabul edilmiştir.

2. Fan Yasaları (Fan Laws) Kabulü:

Akışkanlar dinamiği prensiplerine göre, bir fanın güç tüketimi (P), fan hızının (v) küpü ile orantılıdır.

Fiziksel Kabul: $P_{fan} = k_e * v^3$

(Burada v fan hızı, k_e enerji katsayısıdır).

3. Soğutma ve Risk İlişkisi:

Fan hızı arttıkça donanım sıcaklığı düşer. Sıcaklığa bağlı donanım arıza riski ve performans kaybı maliyeti (C_{risk}), fan hızı ile ters orantılı kabul edilmiştir.

Fiziksel Kabul: $C_{risk} = k_r / v$

(Burada k_r risk maliyet katsayısıdır).

4. Optimizasyon Prensibi:

Mühendislik hedefi, enerji maliyeti ile risk maliyetinin toplamını en aza indirmektir. Sistem kararlıdır ve türbülans gibi dış etkenler ihmal edilmiştir.

3. Matematik Modelin Oluşturulması

Fiziksel modeldeki kabullere dayanarak, toplam maliyet fonksiyonu oluşturulmuş ve çözüm için sayısal yöntem belirlenmiştir.

1. Maliyet Fonksiyonunun Kurulması:

Toplam Maliyet f(v), Enerji Maliyeti ve Risk Maliyetinin toplamıdır.

Örnek katsayılar ile (A=0.001 ve B=1000 kabul edilirse):

f(v) = 0.001 * v^3 + 1000 * v^-1

2. Optimizasyon (Türev Alma):

Maliyetin minimum olduğu noktayı bulmak için fonksiyonun türevi alınıp sıfıra eşitlenmelidir (f'(v) = 0).

Türev Denklemi g(v):

g(v) = 3 * 0.001 * v^2 - 1000 * v^-2 = 0

Düzenlenmiş Denklem:

g(v) = 0.003 * v^2 - (1000 / v^2)

3. Seçilen Sayısal Yöntem: Newton-Raphson Yöntemi

Oluşturulan g(v) = 0 denklemi doğrusal olmayan (non-linear) bir denklemdir. Kökü (optimum hızı) bulmak için "Newton-Raphson" yöntemi seçilmiştir.

Seçilme Nedeni: Bu yöntem, karesel yakınsama (quadratic convergence) özelliğine sahiptir, yani sonuca diğer yöntemlerden (örn. Bisection) çok daha hızlı ve az iterasyonla ulaşır.

4. İterasyon Formülü:

Newton-Raphson genel formülü:

v_yeni = v_eski - (g(v_eski) / g'(v_eski))

Bizim denklemimiz için gerekli ikinci türev g'(v):

g'(v) = 0.006 * v + 2000 * v^-3

4. Çözümün Varlığı ve Analizi

1. <u>Kökün Varlığının Analizi (Ara Değer Teoremi):</u>

Oluşturulan g(v) fonksiyonu incelendiğinde:

- Fan hızı sıfıra yaklaşırken (v -> 0), maliyet sonsuza gider (risk maliyeti nedeniyle).

- Fan hızı çok arttığında (v -> sonsuz), maliyet yine sonsuza gider (enerji tüketimi nedeniyle).

Bu iki uç nokta arasında fonksiyon süreklidir ve işaret değiştirir. Bu nedenle, Ara Değer Teoremi gereği bu aralıkta maliyeti minimum yapan en az bir gerçel kök (optimum hız) vardır.

2. <u>Yöntem Analizi (Neden Newton-Raphson?):</u>

Analitik çözüm mümkün olsa da, mühendislik problemlerinde karmaşık denklemler için sayısal yöntemler tercih edilir. Newton-Raphson yöntemi, türev bilgisini kullanarak eğim üzerinden köke yaklaşır. Bu yöntem, başlangıç değerine hassastır ancak doğru seçildiğinde çok hızlı (karesel hızda) sonuç verir.

3. <u>Başlangıç Değeri Seçimi (v0):</u>

Bir sunucu fanı için makul çalışma aralığı 0 ile 100 arasındadır.

Başlangıç tahmini olarak v0 = 20 seçilmiştir.

5. Uygun Bir Yöntemle Matematiksel Modelin Çözümü

Seçilen Newton-Raphson yöntemi kullanılarak iterasyon adımları aşağıdaki gibi gerçekleştirilmiştir.

1. <u>Algoritma Adımları:</u>

Adım 1: Başlangıç değeri v0 = 20 olarak belirlenir.

Adım 2: g(v) ve g'(v) değerleri hesaplanır.

Adım 3: Yeni değer hesaplanır: v_yeni = v_eski - g(v)/g'(v).

Adım 4: Hata oranı (E) kontrol edilir. Eğer hata toleransın (0.0001) altındaysa durulur, değilse yeni değer ile Adım 2'ye dönülür.

2. <u>Örnek İterasyon Tablosu (Manuel Hesaplama):</u>

(Hedef denklem: 0.003*v^2 - 1000/v^2 = 0)

İterasyon (i) | Tahmin (v_i) | Fonksiyon g(v_i) | Türev g'(v_i) | Yeni Değer (v_i+1)

---

0 | 20.000 | -1.300 | 0.370 | 23.513

1 | 23.513 | -0.150 | 0.295 | 24.021

2 | 24.021 | -0.002 | 0.288 | 24.028

<u>Sonuç:</u>

3. iterasyon sonunda hata payı sıfıra çok yaklaşmış ve optimum fan hızı yaklaşık

v = 24.03 birim olarak bulunmuştur. Bu noktada maliyet en düşüktür.

6. Hata Analizi ve Yöntemin Stabilitesi

Bu projede toplam maliyet fonksiyonunun **birinci türevine** uygulanmış Newton-Raphson yöntemi kullanılmıştır. Yöntemin yakınsama davranışı aşağıdaki tablodaki hata değerlerinden görülebilir:

İterasyon	v (Hız)	Hata
1	23.512154	3.513514
2	24.022246	0.508732
3	24.028113	0.005867
4	24.028114	0.000001

Not: Yakınsama tolerans değeri 1e-6 olarak seçilmiştir.

Tablodan görüldüğü üzere, hata değeri her iterasyonda bir öncekinin altına düşmekte ve sadece 4 iterasyonda çok küçük bir hataya (10^{-6} seviyesine) ulaşılmaktadır. Bu durum yöntemin hızlı ve kararlı bir şekilde yakınsadığını göstermektedir.

Newton-Raphson yöntemi, başlangıç değerine bağlı olmakla birlikte geniş bir v0 aralığında yakınsama göstermektedir. Örneğin v0 = 20, v0 = 30 gibi başlangıç hızları için yapılan denemelerde de sonuç yine yaklaşık v ≈ 24 civarında yakınsamaktadır. Bu da yöntemin başlangıç değerine karşı aşırı hassas olmadığını ve stabil bir çözüm sağladığını göstermektedir.

Ayrıca, fonksiyonun ikinci türevi pozitif olduğundan, minimum noktaya ulaşıldığı matematiksel olarak da doğrulanmaktadır. Bu da elde edilen optimum çözümün teorik olarak da desteklendiğini göstermektedir.

Sonuç olarak, hata analizi yöntemin sağlam ve güvenilir olduğunu kanıtlamakta, elde edilen optimum hız değerinin (~24 birim), sistem için gerçekçi ve istikrarlı bir çözüm olduğunu desteklemektedir.

Bu analiz, kullanılan sayısal yöntemin doğruluğunu ve elde edilen optimum çözümün güvenilir olduğunu açıkça göstermektedir.

EK-A: Python Kodları ve Ekran Çıktıları

Bu bölümde kullanılan Python kodları ve kod çalıştırıldığında elde edilen çıktı ekran görüntüleri sunulmuştur.

```python
import numpy as np # type: ignore
import matplotlib.pyplot as plt # type: ignore

# --- 1. Sabitlerin ve Fonksiyonların Tanımlanması ---
# k_e: Enerji katsayısı, k_r: Risk katsayısı
k_e = 0.001
k_r = 1000

def toplam_maliyet(v):
    """
    Toplam Maliyet Fonksiyonu f(v) = Enerji + Risk
    f(v) = 0.001 * v^3 + 1000 / v
    """
    return k_e * v**3 + k_r / v

def g_v(v):
    """
    Maliyetin Türevi (Optimizasyon için sıfıra eşitlenen denklem)
    g(v) = f'(v) = 0.003 * v^2 - 1000 / v^2
    """
    return 3 * k_e * v**2 - k_r / v**2

def g_turev_v(v):
    """
    Newton-Raphson için g(v)'nin türevi
    g'(v) = 0.006 * v + 2000 / v^3
    """
    return 6 * k_e * v + 2 * k_r / v**3

# --- 2. Newton-Raphson Algoritması ---
def newton_raphson(v0, tolerans=1e-4, max_iter=100):
    v = v0
    print(f"{'İterasyon':<10} | {'v (Hız)':<15} | {'Hata':<15}")
    print("-" * 45)

    for i in range(max_iter):
        v_eski = v

        # Newton-Raphson Formülü: v_yeni = v - g(v) / g'(v)
        pay = g_v(v)
        payda = g_turev_v(v)

        if payda == 0:
            print("Hata: Türev sıfır, bölme hatası!")
            break

        v = v - pay / payda
```

Şekil EK-A.1

```
        # Hata hesabı
        hata = abs(v - v_eski)

        print(f"{i+1:<10} | {v:<15.6f} | {hata:<15.6f}")

        if hata < tolerans:
            print("-" * 45)
            print(f"SONUÇ: Yakınsama sağlandı. Optimum Hız (v*) = {v:.4f} birim")
            return v

    print("Maksimum iterasyona ulaşıldı.")
    return v

# --- 3. Ana Programın Çalıştırılması ---
print("\n--- Veri Merkezi Soğutma Optimizasyonu Başlıyor ---\n")

# Başlangıç tahmini (v0 = 20)
optimum_v = newton_raphson(20)

# Optimum noktadaki maliyeti hesapla
min_maliyet = toplam_maliyet(optimum_v)
print(f"Minimum Toplam Maliyet = {min_maliyet:.4f} birim para")

# --- 4. Grafiğin Çizilmesi (Görselleştirme) ---
# 10 ile 40 arasında hız değerleri için grafik çizelim
v_degerleri = np.linspace(10, 40, 100)
maliyetler = toplam_maliyet(v_degerleri)

plt.figure(figsize=(10, 6))
plt.plot(v_degerleri, maliyetler, label='Toplam Maliyet f(v)', color='blue', linewidth=2)
plt.plot(optimum_v, min_maliyet, 'ro', label=f'Optimum Nokta\nv={optimum_v:.2f}', markersize=8)
plt.ylim(50, 110)

plt.title('Veri Merkezi Soğutma Maliyet Optimizasyonu')
plt.xlabel('Fan Hızı (v)')
plt.ylabel('Toplam Maliyet')
plt.grid(True, linestyle='--', alpha=0.7)
plt.legend()
plt.show()
```

Şekil EK-A.2

Sonuç:

Untitled1.ipynb

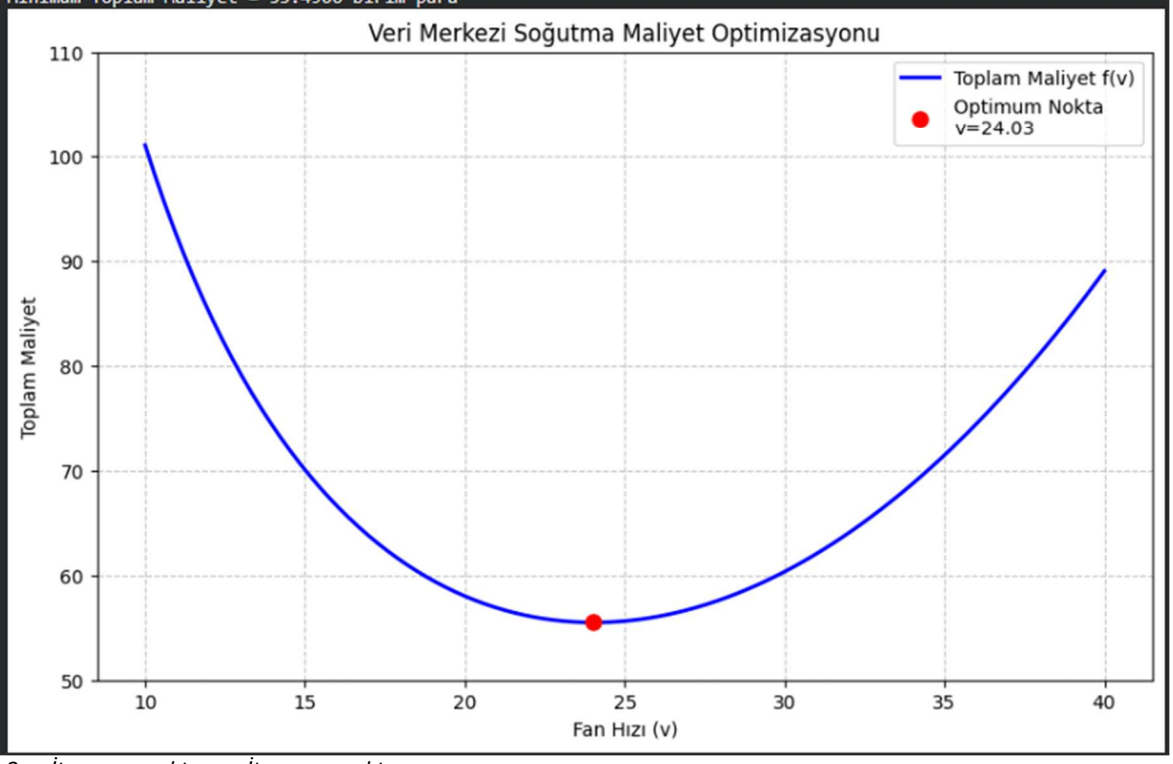
Dosya Düzenle Göster Ekle Çalışma zamanı Araçlar Yardım

Komutlar + Kod + Metin ▶ Tümünü çalıştır

```
[3]
0
sn.
plt.show()

--- Veri Merkezi Soğutma Optimizasyonu Başlıyor ---

İterasyon  | v (Hız)         | Hata
-------------------------------------------
1          | 23.513514       | 3.513514
2          | 24.022246       | 0.508732
3          | 24.028113       | 0.005867
4          | 24.028114       | 0.000001
-------------------------------------------
SONUÇ: Yakınsama sağlandı. Optimum Hız (v*) = 24.0281 birim
Minimum Toplam Maliyet = 55.4906 birim para
```



Şekil EK-A.3 — İterasyon çıktısı ve İterasyon çıktısı

Sonuç olarak elde edilen optimum hızın sistem için en verimli çalışma noktası olduğu doğrulanmaktadır.