**Tugas Metode Numerik**

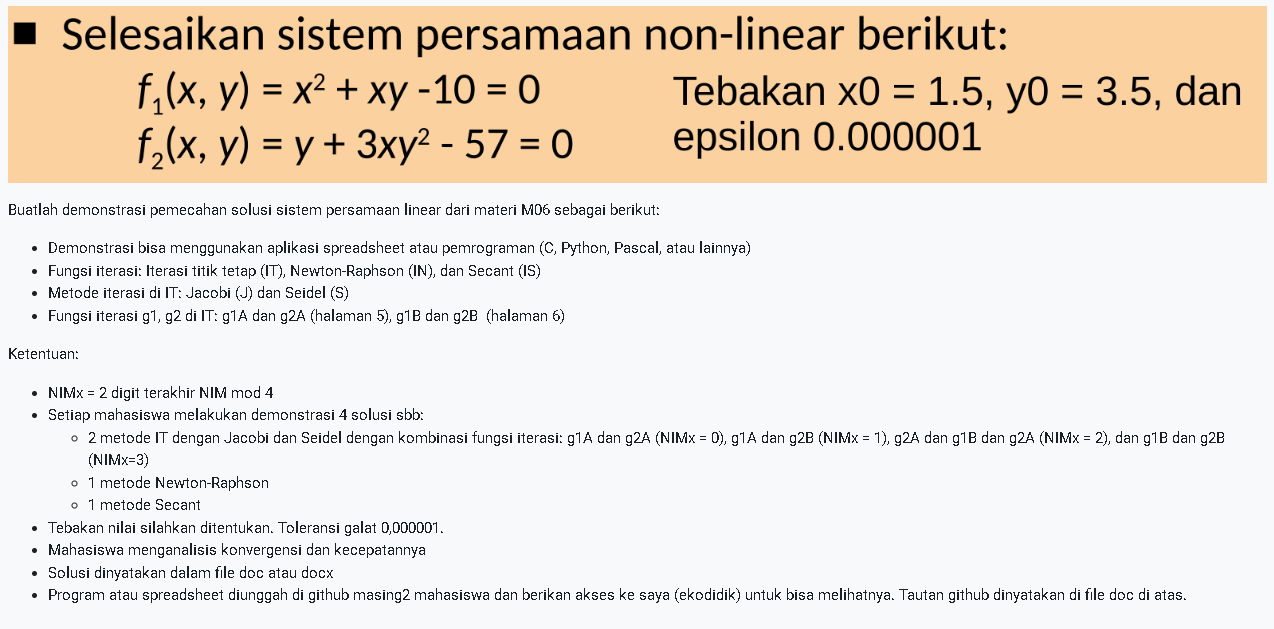
**Sistem Persamaan Nonlinear**

**Nama :** Muhammad Fakhri Akmal Arief

**NIM :** 21120123140172

**Kelas :** Metode Numerik – A

Tugas :



Penyelesaian :

* Demonstrasi menggunakan program yang saya buat menggunakan bahasa pemrograman Python dengan library math.
* 2 digit NIM terakhir saya adalah 72, maka 72 mod 4 = 0
* NIMx = 0, maka kombinasi fungsi iterasi yang saya gunakan adalah g1A dan g2A

Link github : <https://github.com/fakhriakma1a/metode-numerik-1>

Source code

|  |
| --- |
| import math  # Parameter  x0, y0 = 1.5, 3.5  epsilon = 0.000001  max\_iter = 1000  print("="\*80)  print("PENYELESAIAN SISTEM PERSAMAAN NON LINEAR")  print("f1(x,y) = x^2 + xy - 10 = 0")  print("f2(x,y) = y + 3xy^2 - 57 = 0")  print(f"Tebakan awal: x0 = {x0}, y0 = {y0}")  print(f"Epsilon: {epsilon}")  print(f"NIMx = 72 mod 4 = 0")  print("="\*80)  # METODE 1: ITERASI TITIK TETAP - JACOBI dengan g1A dan g2A  print("\n" + "="\*80)  print("METODE 1: ITERASI TITIK TETAP - JACOBI (g1A dan g2A)")  print("="\*80)  print("Fungsi iterasi:")  print("g1A: x = sqrt(10 - xy)")  print("g2A: y = sqrt((57 - y) / (3x))")  print("-"\*80)  def g1A\_jacobi(x, y):  val = 10 - x\*y  if val <= 0:  print(f"Warning: 10 - xy = {val} <= 0 pada x={x:.6f}, y={y:.6f}")  return None  return math.sqrt(val)  def g2A\_jacobi(x, y):  if abs(x) < 1e-10:  return None  val = (57 - y) / (3\*x)  if val <= 0:  print(f"Warning: (57-y)/(3x) = {val} <= 0 pada x={x:.6f}, y={y:.6f}")  return None  return math.sqrt(val)  x, y = x0, y0  print(f"{'Iter':<6} {'x':<15} {'y':<15} {'deltaX':<15} {'deltaY':<15}")  print("-"\*80)  print(f"{0:<6} {x:<15.6f} {y:<15.6f} {0.0:<15.6f} {0.0:<15.6f}")  converged = False  for i in range(1, max\_iter + 1):  x\_old, y\_old = x, y  x\_new = g1A\_jacobi(x\_old, y\_old)  y\_new = g2A\_jacobi(x\_old, y\_old)    if x\_new is None or y\_new is None:  print(f"\nDivergen pada iterasi ke-{i} (nilai di dalam akar negatif)")  break    deltaX = abs(x\_new - x\_old)  deltaY = abs(y\_new - y\_old)    x, y = x\_new, y\_new    if i <= 15 or (deltaX < epsilon and deltaY < epsilon):  print(f"{i:<6} {x:<15.6f} {y:<15.6f} {deltaX:<15.6f} {deltaY:<15.6f}")    if deltaX < epsilon and deltaY < epsilon:  print(f"\nKonvergen pada iterasi ke-{i}")  print(f"Solusi: x = {x:.6f}, y = {y:.6f}")  print(f"Verifikasi: f1({x:.6f}, {y:.6f}) = {x\*\*2 + x\*y - 10:.9f}")  print(f" f2({x:.6f}, {y:.6f}) = {y + 3\*x\*y\*\*2 - 57:.9f}")  converged = True  break  if not converged and x\_new is not None:  print("\nTidak konvergen dalam batas iterasi maksimum")  # METODE 2: ITERASI TITIK TETAP - SEIDEL dengan g1A dan g2A  print("\n" + "="\*80)  print("METODE 2: ITERASI TITIK TETAP - SEIDEL (g1A dan g2A)")  print("="\*80)  print("Fungsi iterasi:")  print("g1A: x = sqrt(10 - xy)")  print("g2A: y = sqrt((57 - y) / (3x)) [menggunakan x\_baru]")  print("-"\*80)  x, y = x0, y0  print(f"{'Iter':<6} {'x':<15} {'y':<15} {'deltaX':<15} {'deltaY':<15}")  print("-"\*80)  print(f"{0:<6} {x:<15.6f} {y:<15.6f} {0.0:<15.6f} {0.0:<15.6f}")  converged = False  for i in range(1, max\_iter + 1):  x\_old, y\_old = x, y    # Hitung x baru  val\_x = 10 - x\_old\*y\_old  if val\_x <= 0:  print(f"\nDivergen pada iterasi ke-{i} (10 - xy = {val\_x} <= 0)")  break  x = math.sqrt(val\_x)    # Gunakan x baru untuk hitung y  if abs(x) < 1e-10:  print(f"\nDivergen pada iterasi ke-{i} (x terlalu kecil)")  break  val\_y = (57 - y\_old) / (3\*x)  if val\_y <= 0:  print(f"\nDivergen pada iterasi ke-{i} ((57-y)/(3x) = {val\_y} <= 0)")  break  y = math.sqrt(val\_y)    deltaX = abs(x - x\_old)  deltaY = abs(y - y\_old)    if i <= 15 or (deltaX < epsilon and deltaY < epsilon):  print(f"{i:<6} {x:<15.6f} {y:<15.6f} {deltaX:<15.6f} {deltaY:<15.6f}")    if deltaX < epsilon and deltaY < epsilon:  print(f"\nKonvergen pada iterasi ke-{i}")  print(f"Solusi: x = {x:.6f}, y = {y:.6f}")  print(f"Verifikasi: f1({x:.6f}, {y:.6f}) = {x\*\*2 + x\*y - 10:.9f}")  print(f" f2({x:.6f}, {y:.6f}) = {y + 3\*x\*y\*\*2 - 57:.9f}")  converged = True  break  if not converged:  print("\nTidak konvergen dalam batas iterasi maksimum")  # METODE 3: NEWTON-RAPHSON  print("\n" + "="\*80)  print("METODE 3: NEWTON-RAPHSON")  print("="\*80)  def f1(x, y):  return x\*\*2 + x\*y - 10  def f2(x, y):  return y + 3\*x\*y\*\*2 - 57  def df1\_dx(x, y):  return 2\*x + y  def df1\_dy(x, y):  return x  def df2\_dx(x, y):  return 3\*y\*\*2  def df2\_dy(x, y):  return 1 + 6\*x\*y  x, y = x0, y0  print(f"{'Iter':<6} {'x':<15} {'y':<15} {'deltaX':<15} {'deltaY':<15}")  print("-"\*80)  print(f"{0:<6} {x:<15.6f} {y:<15.6f} {0.0:<15.6f} {0.0:<15.6f}")  converged = False  for i in range(1, max\_iter + 1):  u = f1(x, y)  v = f2(x, y)    du\_dx = df1\_dx(x, y)  du\_dy = df1\_dy(x, y)  dv\_dx = df2\_dx(x, y)  dv\_dy = df2\_dy(x, y)    det = du\_dx \* dv\_dy - du\_dy \* dv\_dx    if abs(det) < 1e-10:  print(f"Determinan Jacobi terlalu kecil pada iterasi ke-{i}, iterasi dihentikan")  break    x\_new = x - (u \* dv\_dy - v \* du\_dy) / det  y\_new = y + (u \* dv\_dx - v \* du\_dx) / det    deltaX = abs(x\_new - x)  deltaY = abs(y\_new - y)    x, y = x\_new, y\_new    if i <= 20 or (deltaX < epsilon and deltaY < epsilon):  print(f"{i:<6} {x:<15.6f} {y:<15.6f} {deltaX:<15.6f} {deltaY:<15.6f}")    if deltaX < epsilon and deltaY < epsilon:  print(f"\nKonvergen pada iterasi ke-{i}")  print(f"Solusi: x = {x:.6f}, y = {y:.6f}")  print(f"Verifikasi: f1({x:.6f}, {y:.6f}) = {x\*\*2 + x\*y - 10:.9f}")  print(f" f2({x:.6f}, {y:.6f}) = {y + 3\*x\*y\*\*2 - 57:.9f}")  converged = True  break  if not converged:  print("\nTidak konvergen dalam batas iterasi maksimum")  # METODE 4: SECANT  print("\n" + "="\*80)  print("METODE 4: METODE SECANT")  print("="\*80)  print("Catatan: Metode Secant untuk sistem persamaan menggunakan")  print("aproksimasi turunan numerik pada setiap variabel")  print("-"\*80)  # Untuk metode Secant, perlu 2 tebakan awal untuk setiap variabel  x0\_sec, y0\_sec = 1.5, 3.5  x1\_sec, y1\_sec = 1.6, 3.6  print(f"{'Iter':<6} {'x':<15} {'y':<15} {'deltaX':<15} {'deltaY':<15}")  print("-"\*80)  print(f"{0:<6} {x0\_sec:<15.6f} {y0\_sec:<15.6f} {0.0:<15.6f} {0.0:<15.6f}")  print(f"{1:<6} {x1\_sec:<15.6f} {y1\_sec:<15.6f} {abs(x1\_sec-x0\_sec):<15.6f} {abs(y1\_sec-y0\_sec):<15.6f}")  x\_prev, y\_prev = x0\_sec, y0\_sec  x\_curr, y\_curr = x1\_sec, y1\_sec  converged = False  for i in range(2, max\_iter + 1):  f1\_prev = f1(x\_prev, y\_prev)  f1\_curr = f1(x\_curr, y\_curr)  f2\_prev = f2(x\_prev, y\_prev)  f2\_curr = f2(x\_curr, y\_curr)    # Hitung x baru menggunakan aproksimasi secant untuk f1  if abs(f1\_curr - f1\_prev) < 1e-10:  df1\_approx = 1.0  else:  df1\_approx = (x\_curr - x\_prev) / (f1\_curr - f1\_prev)    # Hitung y baru menggunakan aproksimasi secant untuk f2  if abs(f2\_curr - f2\_prev) < 1e-10:  df2\_approx = 1.0  else:  df2\_approx = (y\_curr - y\_prev) / (f2\_curr - f2\_prev)    x\_new = x\_curr - f1\_curr \* df1\_approx  y\_new = y\_curr - f2\_curr \* df2\_approx    deltaX = abs(x\_new - x\_curr)  deltaY = abs(y\_new - y\_curr)    if i <= 20 or (deltaX < epsilon and deltaY < epsilon):  print(f"{i:<6} {x\_new:<15.6f} {y\_new:<15.6f} {deltaX:<15.6f} {deltaY:<15.6f}")    if deltaX < epsilon and deltaY < epsilon:  print(f"\nKonvergen pada iterasi ke-{i}")  print(f"Solusi: x = {x\_new:.6f}, y = {y\_new:.6f}")  print(f"Verifikasi: f1({x\_new:.6f}, {y\_new:.6f}) = {f1(x\_new, y\_new):.9f}")  print(f" f2({x\_new:.6f}, {y\_new:.6f}) = {f2(x\_new, y\_new):.9f}")  converged = True  break    x\_prev, y\_prev = x\_curr, y\_curr  x\_curr, y\_curr = x\_new, y\_new  if not converged:  print("\nTidak konvergen dalam batas iterasi maksimum")  print("\n" + "="\*80)  print("ANALISIS KONVERGENSI")  print("="\*80)  print("\n1. METODE JACOBI vs SEIDEL:")  print(" - Seidel lebih cepat konvergen karena menggunakan nilai x\_baru")  print(" langsung untuk menghitung y")  print(" - Keduanya bergantung pada pemilihan fungsi iterasi yang tepat")  print("\n2. METODE NEWTON-RAPHSON:")  print(" - Konvergensi kuadratik (paling cepat)")  print(" - Memerlukan perhitungan determinan Jacobi dan turunan parsial")  print(" - Lebih stabil untuk berbagai tebakan awal")  print("\n3. METODE SECANT:")  print(" - Aproksimasi Newton-Raphson tanpa turunan eksplisit")  print(" - Konvergensi superlinear (antara linear dan kuadratik)")  print(" - Memerlukan 2 tebakan awal untuk setiap variabel")  print("\n4. PEMILIHAN FUNGSI ITERASI:")  print(" - Fungsi g1A dan g2A: x = sqrt(10-xy), y = sqrt((57-y)/(3x))")  print(" - Konvergen untuk tebakan awal yang dekat dengan solusi")  print(" - Syarat konvergen: |∂g1/∂x| + |∂g1/∂y| < 1 dan")  print(" |∂g2/∂x| + |∂g2/∂y| < 1")  print("="\*80) |

Output

|  |
| --- |
| ================================================================================  PENYELESAIAN SISTEM PERSAMAAN NON LINEAR  f1(x,y) = x^2 + xy - 10 = 0  f2(x,y) = y + 3xy^2 - 57 = 0  Tebakan awal: x0 = 1.5, y0 = 3.5  Epsilon: 1e-06  NIMx = 72 mod 4 = 0  ================================================================================  ================================================================================  METODE 1: ITERASI TITIK TETAP - JACOBI (g1A dan g2A)  ================================================================================  Fungsi iterasi:  g1A: x = sqrt(10 - xy)  g2A: y = sqrt((57 - y) / (3x))  --------------------------------------------------------------------------------  Iter x y deltaX deltaY  --------------------------------------------------------------------------------  0 1.500000 3.500000 0.000000 0.000000  1 2.179449 3.448027 0.679449 0.051973  2 1.576452 2.861895 0.602998 0.586132  3 2.342725 3.383378 0.766273 0.521483  4 1.440026 2.762030 0.902699 0.621348  5 2.454100 3.543284 1.014074 0.781253  6 1.142116 2.694601 1.311984 0.848683  7 2.631056 3.981125 1.488940 1.286524  Warning: 10 - xy = -0.4745606322293927 <= 0 pada x=2.631056, y=3.981125  Divergen pada iterasi ke-8 (nilai di dalam akar negatif)  ================================================================================  METODE 2: ITERASI TITIK TETAP - SEIDEL (g1A dan g2A)  ================================================================================  Fungsi iterasi:  g1A: x = sqrt(10 - xy)  g2A: y = sqrt((57 - y) / (3x)) [menggunakan x\_baru]  --------------------------------------------------------------------------------  Iter x y deltaX deltaY  --------------------------------------------------------------------------------  0 1.500000 3.500000 0.000000 0.000000  1 2.179449 2.860506 0.679449 0.639494  2 1.940534 3.049551 0.238916 0.189045  3 2.020456 2.983405 0.079922 0.066146  4 1.993028 3.005704 0.027428 0.022300  5 2.002385 2.998054 0.009357 0.007650  6 1.999185 3.000666 0.003200 0.002611  7 2.000279 2.999773 0.001094 0.000893  8 1.999905 3.000078 0.000374 0.000305  9 2.000033 2.999973 0.000128 0.000104  10 1.999989 3.000009 0.000044 0.000036  11 2.000004 2.999997 0.000015 0.000012  12 1.999999 3.000001 0.000005 0.000004  13 2.000000 3.000000 0.000002 0.000001  14 2.000000 3.000000 0.000001 0.000000  Konvergen pada iterasi ke-14  Solusi: x = 2.000000, y = 3.000000  Verifikasi: f1(2.000000, 3.000000) = -0.000000816  f2(2.000000, 3.000000) = 0.000000487  ================================================================================  METODE 3: NEWTON-RAPHSON  ================================================================================  Iter x y deltaX deltaY  --------------------------------------------------------------------------------  0 1.500000 3.500000 0.000000 0.000000  1 2.036029 2.843875 0.536029 0.656125  2 1.998701 3.002289 0.037328 0.158413  3 2.000000 2.999999 0.001299 0.002289  4 2.000000 3.000000 0.000000 0.000001  Konvergen pada iterasi ke-4  Solusi: x = 2.000000, y = 3.000000  Verifikasi: f1(2.000000, 3.000000) = 0.000000000  f2(2.000000, 3.000000) = 0.000000000  ================================================================================  METODE 4: METODE SECANT  ================================================================================  Catatan: Metode Secant untuk sistem persamaan menggunakan  aproksimasi turunan numerik pada setiap variabel  --------------------------------------------------------------------------------  Iter x y deltaX deltaY  --------------------------------------------------------------------------------  0 1.500000 3.500000 0.000000 0.000000  1 1.600000 3.600000 0.100000 0.100000  2 1.804878 3.477377 0.204878 0.122623  3 1.883562 3.943536 0.078684 0.466159  4 1.830317 3.233740 0.053245 0.709796  5 1.853126 3.150546 0.022808 0.083194  6 6.463432 3.102769 4.610306 0.047777  7 1.916947 3.151030 4.546485 0.048261  8 1.941806 3.152242 0.024859 0.001211  9 1.957057 3.146021 0.015251 0.006221  10 1.959134 3.267248 0.002077 0.121227  11 1.957164 3.037465 0.001970 0.229783  12 1.958118 3.032008 0.000954 0.005457  13 1.904242 3.030896 0.053876 0.001111  14 1.991109 3.031982 0.086866 0.001085  15 1.990890 3.031561 0.000218 0.000421  16 1.990969 3.013656 0.000079 0.017904  17 1.990888 3.006618 0.000080 0.007039  18 1.991167 3.006671 0.000279 0.000053  19 1.997745 3.006629 0.006578 0.000042  20 1.998108 3.006673 0.000364 0.000044  37 1.999983 3.000060 0.000000 0.000000  Konvergen pada iterasi ke-37  Solusi: x = 1.999983, y = 3.000060  Verifikasi: f1(1.999983, 3.000060) = -0.000000273  f2(1.999983, 3.000060) = 0.001766166  ================================================================================  ANALISIS KONVERGENSI  ================================================================================  1. METODE JACOBI vs SEIDEL:  - Seidel lebih cepat konvergen karena menggunakan nilai x\_baru  langsung untuk menghitung y  - Keduanya bergantung pada pemilihan fungsi iterasi yang tepat  2. METODE NEWTON-RAPHSON:  - Konvergensi kuadratik (paling cepat)  - Memerlukan perhitungan determinan Jacobi dan turunan parsial  - Lebih stabil untuk berbagai tebakan awal  3. METODE SECANT:  - Aproksimasi Newton-Raphson tanpa turunan eksplisit  - Konvergensi superlinear (antara linear dan kuadratik)  - Memerlukan 2 tebakan awal untuk setiap variabel  4. PEMILIHAN FUNGSI ITERASI:  - Fungsi g1A dan g2A: x = sqrt(10-xy), y = sqrt((57-y)/(3x))  - Konvergen untuk tebakan awal yang dekat dengan solusi  - Syarat konvergen: |∂g1/∂x| + |∂g1/∂y| < 1 dan  |∂g2/∂x| + |∂g2/∂y| < 1  ================================================================================ |